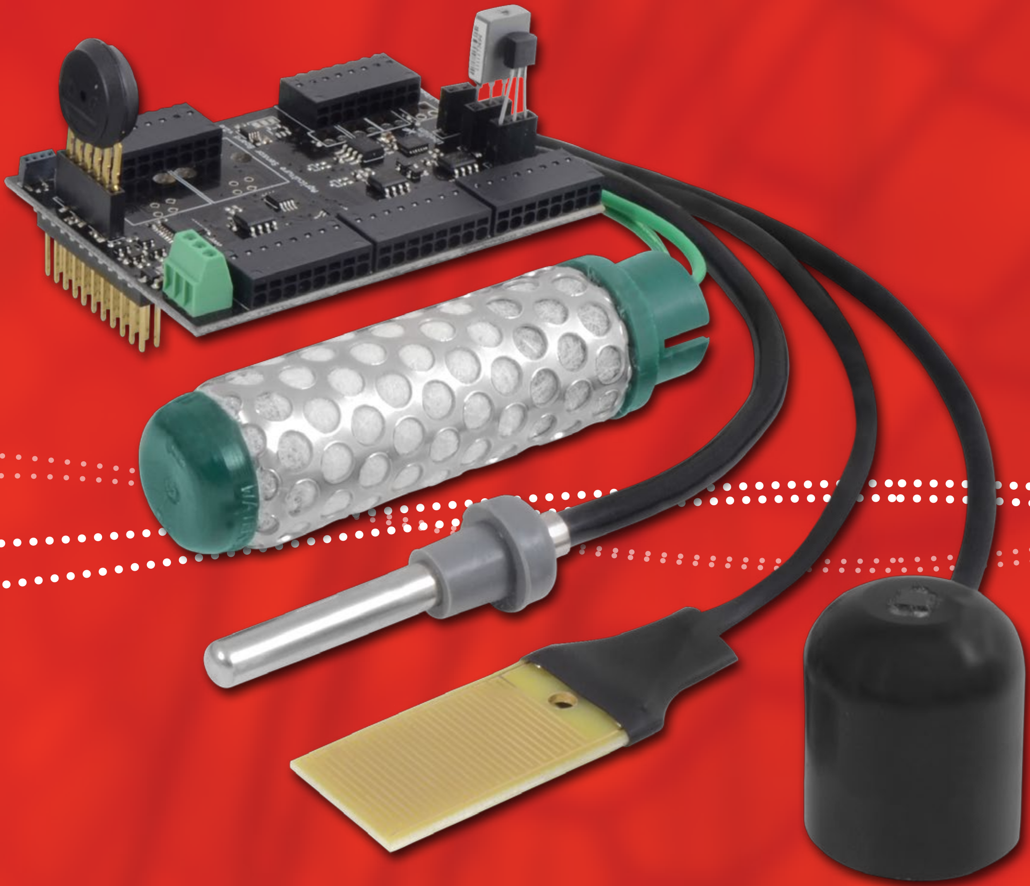
Agriculture 2.0

Technical Guide



libelium

waspmote

Document version: v4.1 - 04/2013

INDEX

1. General

1.1. General and safety information

- In this section, the term "Waspmote" encompasses both the Waspmote device itself and its modules and sensor boards.
- Read through the document "General Conditions of Libelium Sale and Use".
- Do not allow contact of metallic objects with the electronic part to avoid injuries and burns.
- NEVER submerge the device in any liquid.
- Keep the device in a dry place and away from any liquid which may spill.
- Waspmote consists of highly sensitive electronics which is accessible to the exterior, handle with great care and avoid bangs or hard brushing against surfaces.
- Check the product specifications section for the maximum allowed power voltage and amperage range and consequently always use a current transformer and a battery which works within that range. Libelium is only responsible for the correct operation of the device with the batteries, power supplies and chargers which it supplies.
- Keep the device within the specified range of temperatures in the specifications section.
- Do not connect or power the device with damaged cables or batteries.
- Place the device in a place only accessible to maintenance personnel (a restricted area).
- Keep children away from the device in all circumstances.
- If there is an electrical failure, disconnect the main switch immediately and disconnect that battery or any other power supply that is being used.
- If using a car lighter as a power supply, be sure to respect the voltage and current data specified in the "Power Supplies" section.
- If using a battery in combination or not with a solar panel as a power supply, be sure to use the voltage and current data specified in the "Power supplies" section.
- If a software or hardware failure occurs, consult the Libelium Web **Development section**.
- Check that the frequency and power of the communication radio modules together with the integrated antennas are allowed in the area where you want to use the device.
- Waspmote is a device to be integrated in a casing so that it is protected from environmental conditions such as light, dust, humidity or sudden changes in temperature. The board supplied "as is" is not recommended for a final installation as the electronic components are open to the air and may be damaged.

1.2. Conditions of use

- Read the "General and Safety Information" section carefully and keep the manual for future consultation.
- Use Waspmote in accordance with the electrical specifications and the environment described in the "Electrical Data" section of this manual.
- Waspmote and its components and modules are supplied as electronic boards to be integrated within a final product. This product must contain an enclosure to protect it from dust, humidity and other environmental interactions. In the event of outside use, this enclosure must be rated at least IP-65.
- Do not place Waspmote in contact with metallic surfaces; they could cause short-circuits which will permanently damage it.

Further information you may need can be found at: **http://www.libelium.com/development/waspmote**

The "General Conditions of Libelium Sale and Use" document can be found at:

http://www.libelium.com/development/waspmote/technical_service

2. Waspmote Plug & Sense!

The new Waspmote Plug & Sense! line allows you to easily deploy wireless sensor networks in an easy and scalable way ensuring minimum maintenance costs. The new platform consists of a robust waterproof enclosure with specific external sockets to connect the sensors, the solar panel, the antenna and even the USB cable in order to reprogram the node. It has been specially designed to be scalable, easy to deploy and maintain.

***Note:** For a complete reference guide download the "Waspmote Plug & Sense! Technical Guide" in the **Development section** of the **Libelium website**.*

2.1. Features

- Robust waterproof IP65 enclosure
- Add or change a sensor probe in seconds
- Solar powered with internal and external panel options
- Radios available: Zigbee, 802.15.4, Wifi, 868MHz, 900MHz and 3G/GPRS
- Over the air programming (OTAP) of multiple nodes at once
- Special holders and brackets ready for installation in street lights and building fronts
- Graphical and intuitive programming interface

2.2. Sensor Probes

Sensor probes can be easily attached by just screwing them into the bottom sockets. This allows you to add new sensing capabilities to existing networks just in minutes. In the same way, sensor probes may be easily replaced in order to ensure the lowest maintenance cost of the sensor network.

Figure 1: Connecting a sensor probe to Waspmote Plug & Sense!

2.3. Solar Powered

Battery can be recharged using the internal or external solar panel options.
The external solar panel is mounted on a 45º holder which ensures the maximum performance of each outdoor installation.

Figure 2: Waspmote Plug & Sense! powered by an external solar panel

For the internal option, the solar panel is embedded on the front of the enclosure, perfect for use where space is a major challenge.

Figure 3: Internal solar panel

Figure 4: Waspmote Plug & Sense! powered by an internal solar panel

2.4. Programming the Nodes

Waspmote Plug & Sense! can be reprogrammed in two ways:

The basic programming is done from the USB port. Just connect the USB to the specific external socket and then to the computer to upload the new firmware.

Figure 5: Programming a node

Over the Air Programming is also possible once the node has been installed. With this technique you can reprogram wirelessly one or more Waspmote sensor nodes at the same time by using a laptop and the Waspmote Gateway.

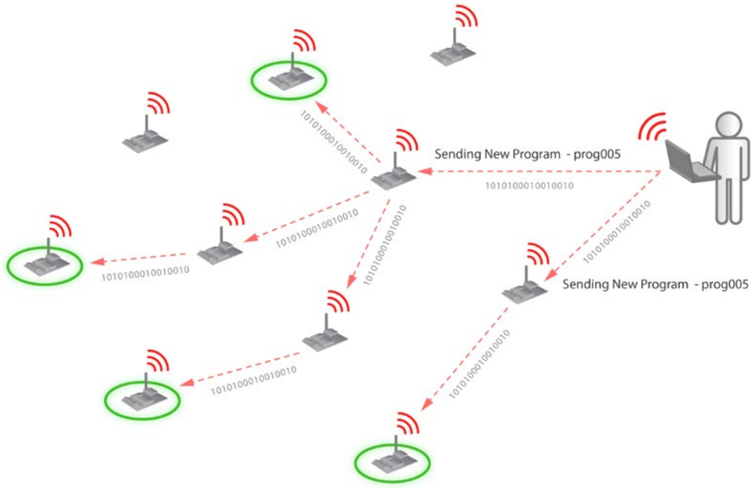


Figure 6: Typical OTAP process

2.5. Radio Interfaces

Model	Protocol	Frequency	txPower	Sensitivity	Range *
XBee-802.15.4-Pro	802.15.4	2.4GHz	100mW	-100dBm	7000m
XBee-ZB-Pro	ZigBee-Pro	2.4GHz	50mW	-102dBm	7000m
XBee-868	RF	868MHz	315mW	-112dBm	12km
XBee-900	RF	900MHz	50mW	-100dBm	10Km
Wifi	802.11b/g	2.4GHz	0dBm - 12dBm	-83dBm	50m-500m
GPRS	-	850MHz/900MHz/ 1800MHz/1900MHz	2W(Class4) 850MHz/900MHz, 1W(Class1) 1800MHz/1900MHz	-109dBm	
3G/GPRS	-	Tri-Band UMTS 2100/1900/900MHz Quad-Band GSM/EDGE, 850/900/1800/1900 MHz	UMTS 900/1900/2100 0,25W GSM 850MHz/900MHz 2W DCS1800MHz/PCS1900MHz 1W	-106dBm	

** Line of sight, Fresnel zone clearance and 5dBi dipole antenna.*

2.6. Program in minutes

In order to program the nodes an intuitive graphic interface has been developed. Developers just need to fill a web form in order to obtain the complete source code for the sensor nodes. This means the complete program for an specific application can be generated just in minutes. Check the Code Generator to see how easy it is at:

http://www.libelium.com/development/plug_&_sense/sdk_and_applications/code_generator

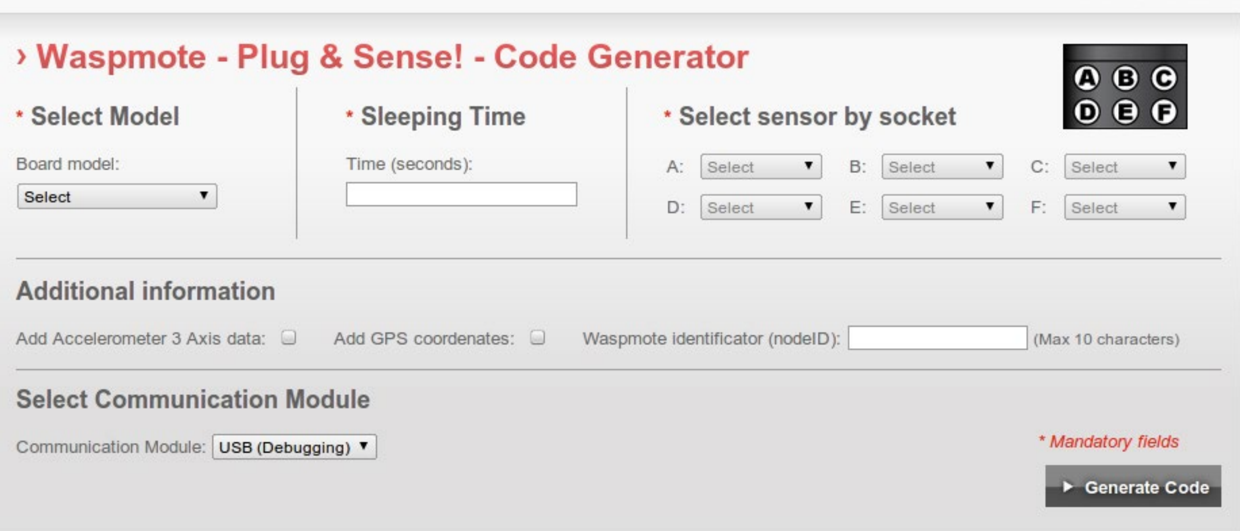


Figure 7: Code Generator

2.7. Data to the Cloud

The Sensor data gathered by the Waspmote Plug & Sense! nodes is sent to the Cloud by **Meshlium**, the Gateway router specially designed to connect Waspmote sensor networks to the Internet via Ethernet, Wifi and 3G interfaces.

Thanks to Meshlium's new feature, the Sensor Parser, now it is easier to receive any frame, parse it and store the data into a local or external Data Base.

Figure 8: Meshlium

2.8. Meshlium Storage Options

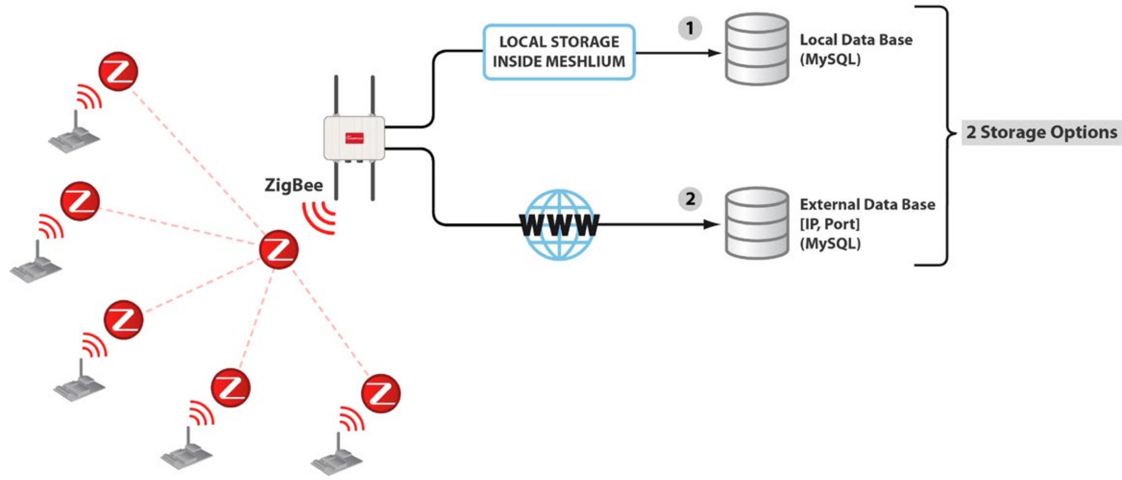


Figure 9: Meshlium Storage Options

- Local Data Base
- External Data Base

2.9. Meshlium Connection Options

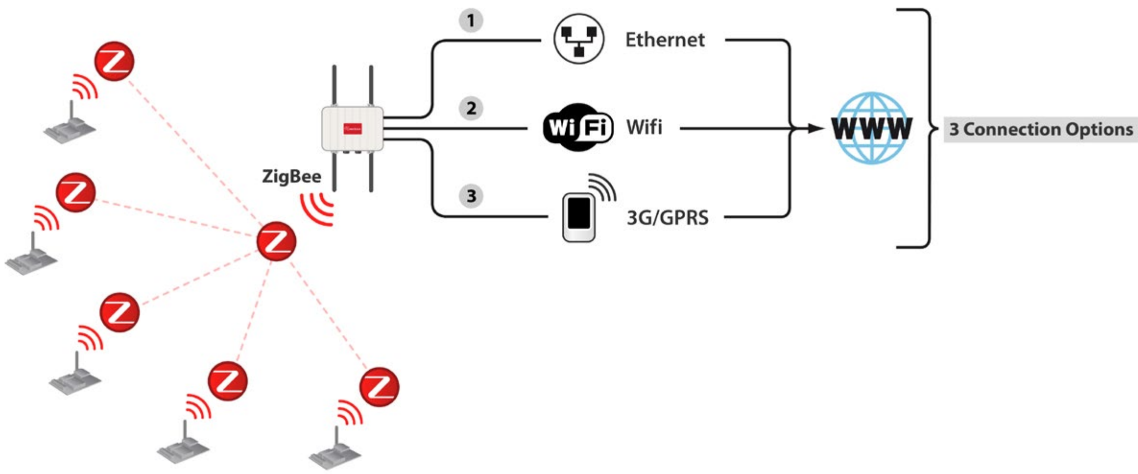


Figure 10: Meshlium Connection Options

- ZigBee → Ethernet
- ZigBee → Wifi
- ZigBee → 3G/GPRS

2.10. Models

There are some defined configurations of Waspmote Plug & Sense! depending on which sensors are going to be used. Waspmote Plug & Sense! configurations allow to connect up to six sensor probes at the same time.

Each model takes a different conditioning circuit to enable the sensor integration. For this reason each model allows to connect just its specific sensors.

This section describes each model configuration in detail, showing the sensors which can be used in each case and how to connect them to Waspmote. In many cases, the sensor sockets accept the connection of more than one sensor probe. See the compatibility table for each model configuration to choose the best probe combination for the application.

It is very important to remark that each socket is designed only for one specific sensor, so **they are not interchangeable**. Always be sure you connected probes in the right socket, otherwise they can be damaged.

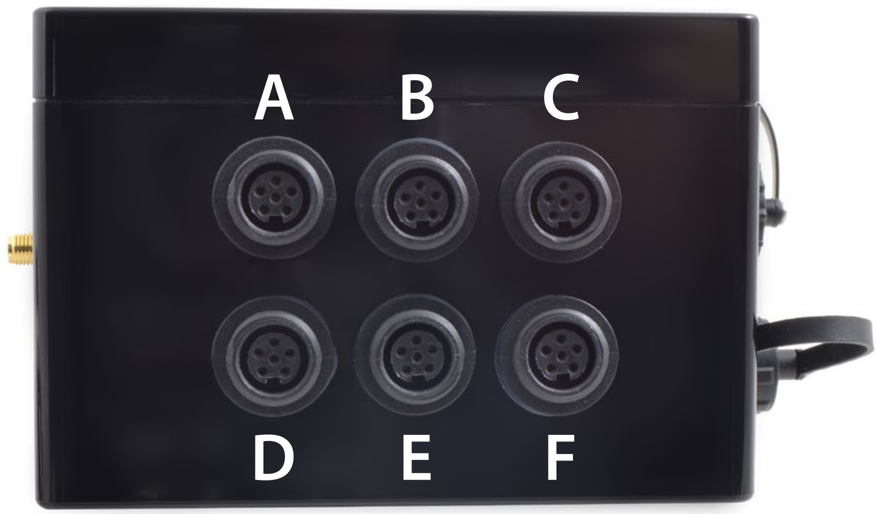


Figure 11: Identification of sensor sockets

2.10.1. Smart Agriculture

The Smart Agriculture models allow to monitor multiple environmental parameters involving a wide range of applications. It has been provided with sensors for air and soil temperature and humidity (Sensirion), solar visible radiation, wind speed and direction, rainfall, atmospheric pressure , etc.

The main applications for this Waspmote Plug & Sense! model are precision agriculture, irrigation systems, greenhouses, weather stations, etc. Refer to **Libelium website** for more information.

Two variants are possible for this model, normal and PRO. Next section describes each configuration in detail.

Figure 12: Smart Agriculture Waspmote Plug & Sense! model

Normal

Sensor sockets are configured as shown in the figure below.

Sensor Socket	Sensor probes allowed for each sensor socket	
	Parameter	Reference
A	Humidity + Temperature (Sensirion)	9247
B	Atmospheric pressure	9250
C	Soil temperature	86949*
	Soil moisture	9248
D	Weathermeters + pluviometer	9256
E	Soil moisture	9248
F	Leaf wetness	9249
	Soil moisture	9248

Figure 13: Sensor sockets configuration for Smart Agriculture model

* Ask Libelium **Sales Department** for more information.

Note: *For more technical information about each sensor probe go to the* ***Development section*** *in Libelium website.*

PRO

Sensor sockets are configured as shown in the figure below.

Sensor Socket	Sensor probes allowed for each sensor socket	
	Parameter	Reference
A	Humidity + Temperature (Sensirion)	9247
B	Soil temperature	9255
C	Solar radiation	9251, 9257
D	Soil temperature	86949*
	Soil moisture	9248
E	Dendrometers	9252, 9253, 9254
	Soil moisture	9248
F	Lear wetness	9249
	Soil moisture	9248

Figure 14: Sensor sockets configuration for Smart Agriculture PRO model

* Ask Libelium **Sales Department** for more information.

Note: *For more technical information about each sensor probe go to the* ***Development section*** *in Libelium website.*

3. Hardware

3.1. General Description

The Waspmote Agriculture 2.0 Board allows to monitor **multiple environmental parameters** involving a wide range of applications, from growing development analysis to weather observation. For this, it has been provided with sensors for air and soil temperature and humidity, luminosity, solar visible radiation, wind speed and direction, rainfall, atmospheric pressure, leaf wetness and fruit or trunk diameter (dendrometer). Up to **15 sensors can be connected at the same time**. With the objective of extending the durability of the device after the deployment, the board is endowed with a solid state switches system that facilitates a precise regulation of its power, prolonging the life of the battery.

3.2. Specifications

Weight: 20gr
Dimensions: 73.5 x 51 x 1.3 mm
Temperature Range: [-20ºC, 65ºC]

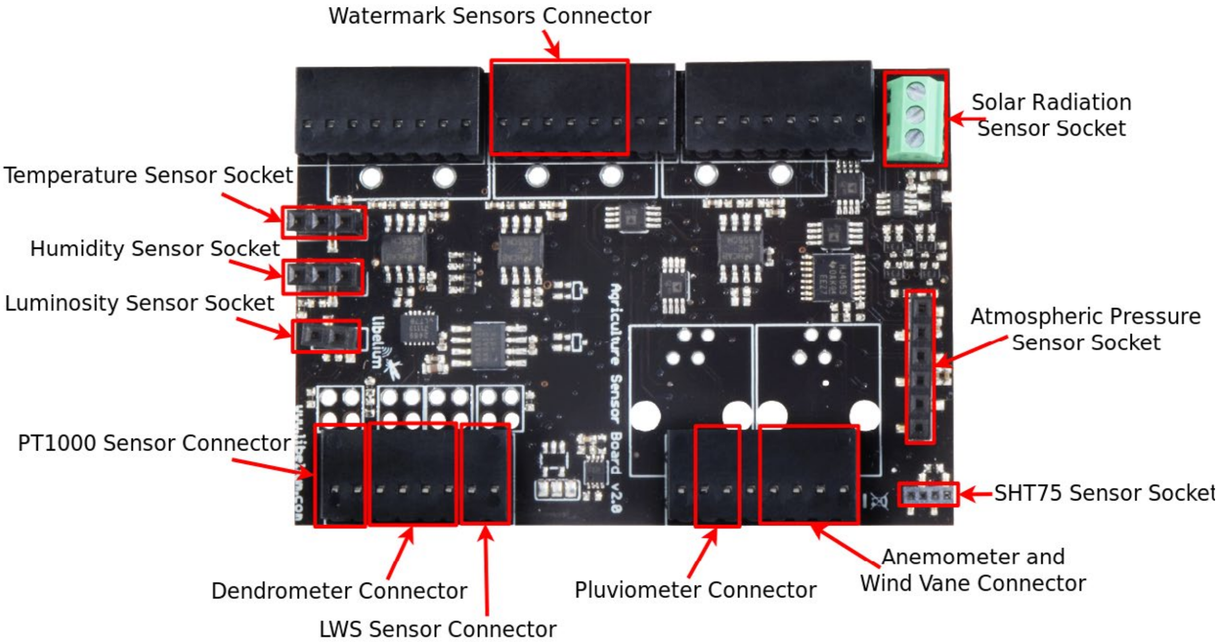


Figure 15: Upper side

3.3. Electrical Characteristics

Board Power Voltages: 3.3V & 5V
Sensor Power Voltages: 3.3V & 5V
Maximum admitted current (continuous): 200mA
Maximum admitted current (peak): 400mA

3.4. Agriculture 2.0 Board Versions

The Agriculture 2.0 Board for Waspmote includes all the electronics and sockets necessary to connect the most typical sensors in agriculture applications, air temperature and humidity (MCP9700A, 808H5V5 and SHT75), soil moisture (Watermark), leaf wetness (LWS), atmospheric pressure (MPX4115A) and weather station (pluviometer, anemometer and vane). A PRO version of the board has been created, which also includes the components necessary to include sensors for more specific applications, such as the solar radiation sensor (SQ-110 or SU-100), the dendrometers (DD, DC2 and DF), and the soil temperature sensor (PT1000).

Sensors in the Agriculture 2.0 Board:

- Temperature sensor MCP9700A by Microchip
- Humidity sensor 808H5V5 by Sencera
- Temperature and humidity sensor SHT75 by Sensirion
- Soil moisture sensor Watermark by Irrometer
- Atmospheric pressure sensor MPX4115A by Freescale
- Leaf wetness sensor LWS
- Weather Station (Anemometer, Wind Vane and Pluviometer)
- Luminosity sensor (LDR)

Sensors added in the PRO version:

- Solar radiation sensor SQ-110 by Apogee
- Ultraviolet radiation sensor SU-100 by Apogee
- DC2, DD and DF dendrometers by Ecomatik
- Soil temperature sensor PT1000

4. Sensors

4.1. Atmospheric Pressure Sensor (MPX4115A)

4.1.1. Specifications

Measurement range: 15 ~ 115kPa
Output signal: 0.2 ~ 4.8V (0 ~ 85ºC)
Sensitivity: 46mV/kPa
Accuracy: <±1.5%V (0 ~ 85ºC)
Typical consumption: 7mA
Maximum consumption: 10mA
Supply voltage: 4.85 ~ 5.35V
Operation temperature: -40 ~ +125ºC
Storage temperature: -40 ~ +125ºC
Response time: 20ms

Figure 16: MPX4115ASensor

4.1.2. Measurement Process

The MPX4115A sensor converts atmospheric pressure to an analog voltage value in a range covering between 0.2V and 4.8V. As this is a range which exceeds the maximum value admitted by Waspmote, its output has been adapted to fit in a range between 0.12V and 2.88V.

To read the sensor it is sufficient to capture the analog value in its input (`ANALOG3`) via the corresponding command. The function of the library `readValue` returns the atmospheric pressure value in kilopascals (kPa). The 5V power supply of the sensor may be connected or disconnected through a switch by activating or deactivating the digital pin `DIGITAL7`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

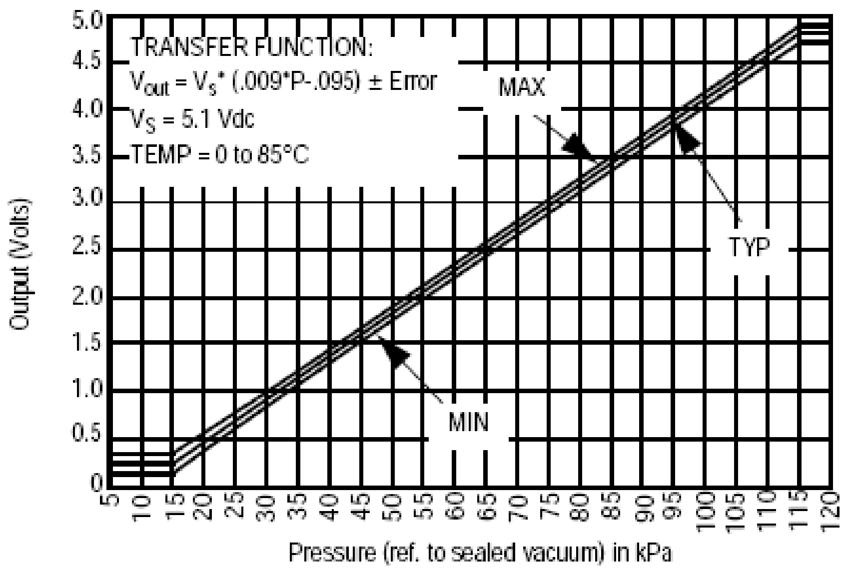


Figure 17: Graph of the MPX4115A sensor's output voltage with regard to pressure taken from the Freescale sensor's data sheet

Reading Code:

```
{
  float value_pressure = 0;
  SensorAgrv20.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_PRESSURE);
  delay(100);  //waiting for the stabilization of the power supply
  value_pressure = SensorAgrv20.readValue(SENS_AGR_PRESSURE);
}
```

You can find a complete example code for reading the atmospheric pressure sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-3-pressure-sensor-reading

4.1.3. Socket

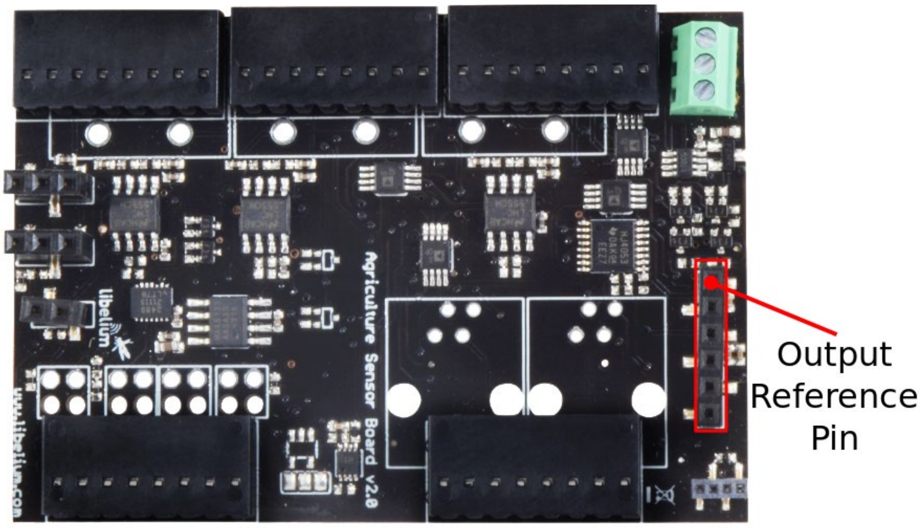


Figure 18: Image of the socket for the MPX4115A sensor

A socket consisting of a 6 female pins, 2.54mm pitch strip has been added for the atmospheric pressure sensor. More information about the sockets for casing applications can be found in section "Sockets for Casing".

4.2. Leaf Wetness Sensor (LWS)

4.2.1. Specifications

Resistance Range: 5kΩ ~ >2MΩ
Output Voltage Range: 1V ~ 3.3V
Length: 3.95cm
Width: 1.95 cm

Figure 19: Leaf Wetness Sensor

4.2.2. Measurement Process

The leaf wetness sensor behaves as a resistance of a very high value (infinite, for practical purposes) in absence of condensation in the conductive combs that make it up, and that may fall down to about 5kΩ when it is completely submerged in water. The voltage at its output is inversely proportional to the humidity condensed on the sensor, and can be read at the analog input of Waspmote ANALOG6 when selected the proper output of a multiplexer that connects this sensor and one of the Watermark soil humidity sensors to that analog pin. The value returned by the reading function of the library corresponds to the percentage of condensation present on the sensor. The power supply of the sensor (3.3V) can be cut off or connected through the switched controlled by the digital pin DIGITAL5. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

Reading code:

```
{
  float value_lw = 0;
  SensorAgrv20.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_LEAF_WETNESS);
  delay(100);  //waiting for the stabilization of the power supply
  value_lw = SensorAgrv20.readValue(SENS_AGR_LEAF_WETNESS);
}
```

You can find a complete example code for reading the leaf wetness sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-5-leaf-wetness-sensor-reading

4.2.3. Socket

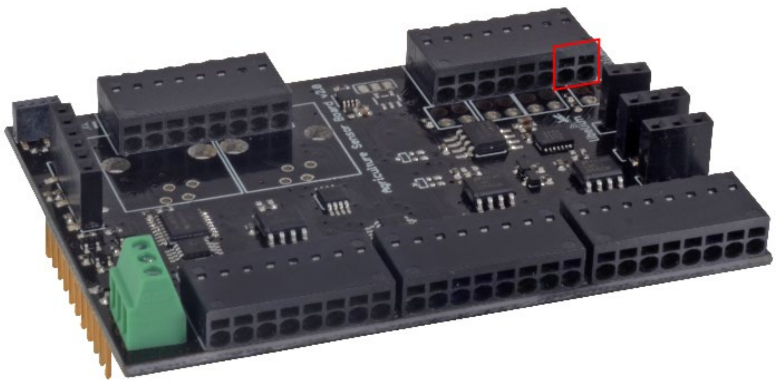


Figure 20: Image of the socket for the Leaf Wetness Sensor

In the image of the figure 20 we can see highlighted the two pins of the terminal block where the sensor must be connected to the board.

4.3. Humidity Sensor (808H5V5)

4.3.1. Specifications

Measurement range: 0 ~ 100%RH
Output signal: 0.8 ~ 3.9V (25ºC)
Accuracy: <±4%RH (a 25ºC, range 30 ~ 80%), <±6%RH (range 0 ~ 100)
Typical consumption: 0.38mA
Maximum consumption: 0.5mA
Power supply: 5VDC ±5%
Operation temperature: -40 ~ +85ºC
Storage temperature: -55 ~ +125ºC
Response time: <15 seconds

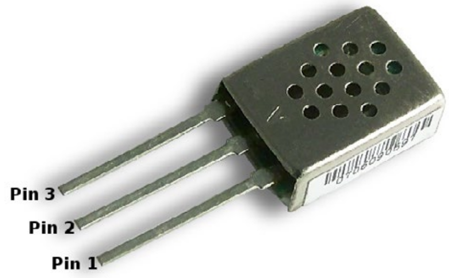


Figure 21: Image of the 808H5V5 sensor

4.3.2. Measurement Process

This is an analog sensor which provides a voltage output proportional to the relative humidity in the atmosphere. As the sensor's signal is outside of that permitted to the Waspmote's input, it's output voltage has been adapted to a range of values between 0.48V and 2.34V.

The reading of the sensor is carried out at the analog input pin `ANALOG2`. The function of the library `readValue` returns the humidity value in relative humidity percentage (%RH). Like the output of the Leaf Wetness Sensor, the output of the 808H5V5 is connected, along with the output of one of the Watermark sensors, to a multiplexer that allows the connection of both sensors at the same time. The output of this multiplexer can be selected using the digital pin `DIGITAL3`, while the 5V power of the sensor is regulated through a switch activated by the digital pin `DIGITAL5`, that controls the power of the solar radiation, temperature, humidity, dendrometer, leaf wetness and temperature plus humidity sensors simultaneously. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

Reading code:

```
{
 float value_humidity = 0;
 SensorAgrv20.ON();
 SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_HUMIDITY);
 delay(15000); //waiting for the sensor response time
 value_humidity = SensorAgrv20.readValue(SENS_AGR_HUMIDITY);
}
```

You can find a complete example code for reading the humidity sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-2-humidity-sensor-reading

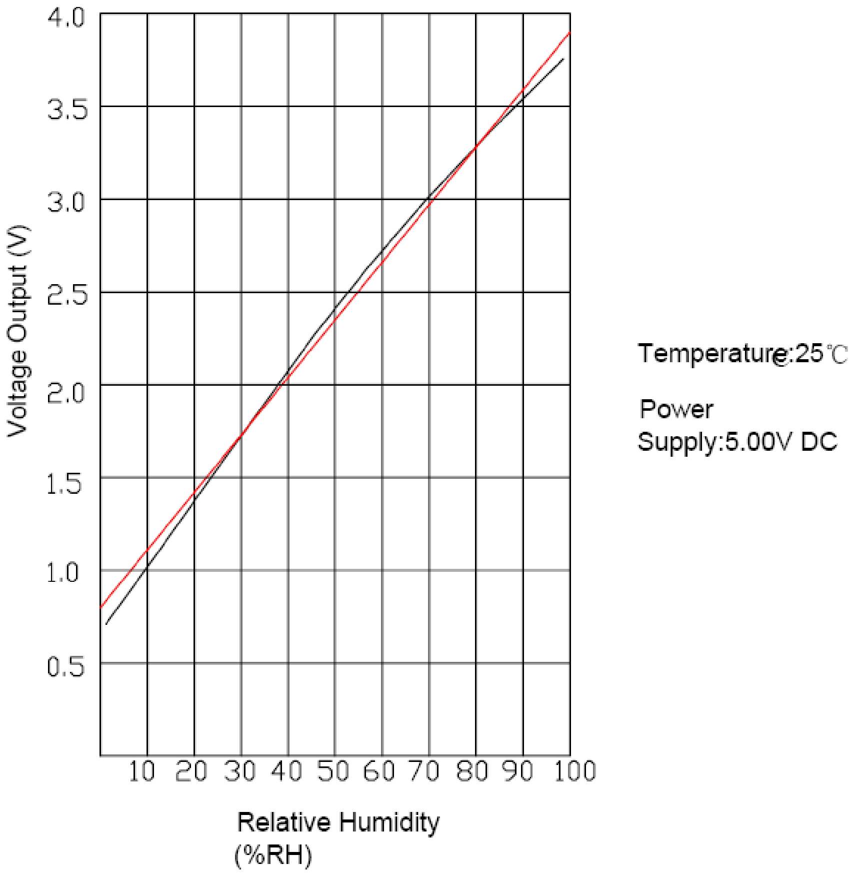


Figure 22: 808H5V5 humidity sensor output taken from the Sencera Co. Ltd sensor data sheet

4.3.3. Socket

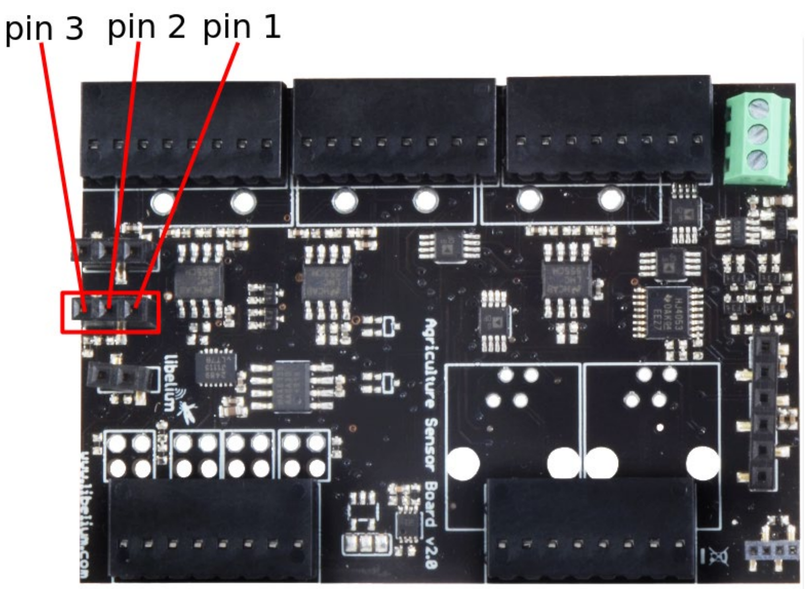


Figure 23: Image of the socket for the 808H5V5 sensor

In figure 23 we have an image of the socket for the 808H5V5 sensor with the pins corresponding to the ones marked in the image in figure 21. More information about the sockets for casing applications can be found in section "Sockets for Casing".

4.4. Temperature Sensor (MCP9700A)

4.4.1. Specifications

Measurement range: -40ºC ~ +125ºC
Output voltage (0ºC): 500mV
Sensitivity: 10mV/ºC
Accuracy: ±2ºC (range 0ºC ~ +70ºC), ±4ºC (range -40 ~ +125ºC)
Typical consumption: 6μA
Maximum consumption: 12μA
Power supply: 2.3 ~ 5.5V
Operation temperature: -40 ~ +125ºC
Storage temperature: -65 ~ 150ºC
Response time: 1.65 seconds (63% of the response for a range from +30 to +125ºC)

Figure 24: Image of the MCP9700A temperature sensor

4.4.2. Measurement Process

The MCP9700A is an analog sensor which converts a temperature value into a proportional analog voltage. The range of output voltages is between 100mV (-40ºC) and 1.75V (125ºC), resulting in a variation of 10mV/ºC, with 500mV of output for 0ºC. The output can thus be read directly from Waspmote through the capture command of the value of the analog pin `ANALOG4`, to which it is connected through a multiplexer shared with one of the Watermark sensors and controlled by the digital pin `DIGITAL3`. The function of the library `readValue` returns the temperature value in Celsius degree (ºC).The 5V power supply is regulated through a digital switch that can be activated or disconnected using the digital pin `DIGITAL5`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

Reading code:

```
{
  float value_temperature = 0;
  SensorAgrv20.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_TEMPERATURE);
  delay(100);   //waiting for the sensor's response time
  value_temperature = SensorAgrv20.readValue(SENS_AGR_TEMPERATURE);
}
```

You can find a complete example code for reading the temperature sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-1-temperature-sensor-reading

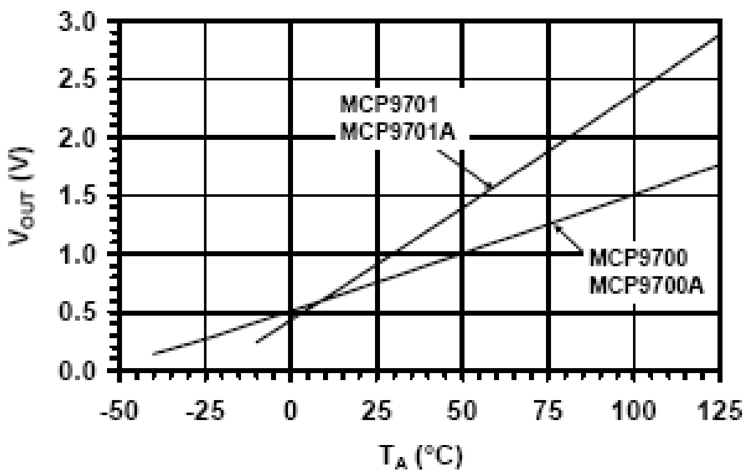


Figure 25: Graph of the MCP9700A sensor output voltage with respect to temperature, taken from the Microchip sensor's data sheet

4.4.3. Socket

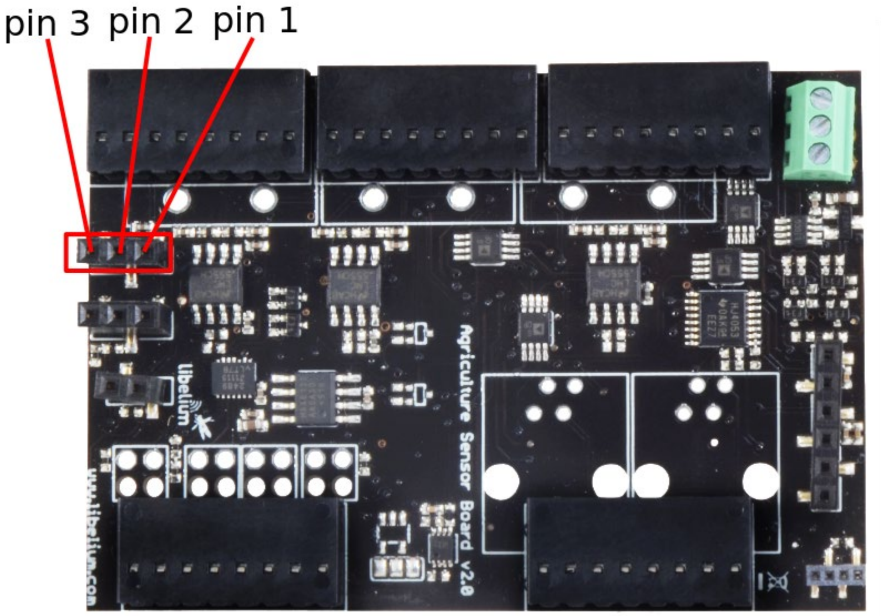


Figure 26: Image of the socket for the MCP900A sensor

The socket for the MCP9700A sensor is basically composed of a three female pins, 2.54mm pitch strip which provides ground, 5V power supply and signal connection to the sensor. In the image in figure 26 we can see marked the pins of the socket that correspond to those in image 24. More information about the sockets for casing applications can be found in section "Sockets for Casing".

4.5. Luminosity Sensor (LDR)

4.5.1. Specifications

Resistance in darkness: 20MΩ
Resistance in light (10lux): 5 ~ 20kΩ
Spectral range: 400 ~ 700nm
Operating Temperature: -30ºC ~ +75ºC
Minimum consumption: 0uA approximately

Figure 27: Image of the LDR luminosity sensor

4.5.2. Measurement Process

This is a resistive sensor whose conductivity varies depending on the intensity of light received on its photosensitive part. The measurement of the sensor is carried out through the analog-to-digital converter of the mote, reading the resulting voltage out of a voltage divider formed by the sensor itself and the load resistor of the socket upon which it has been connected (10kΩ) in the analog input pin `ANALOG7`. This sensor shares the power supply with the dendrometer, PT100, digital humidity and temperature sensor and humidity, temperature, leaf wetness and solar radiation sensors, which can be controlled through the output digital pin `DIGITAL5`. The function of the library `readValue` returns its output value as the voltage formed in the voltage divider.

The measurable spectral range (400nm – 700nm) coincides with the human visible spectrum so it can be used to detect light/ darkness in the same way that a human eye would detect it.

Below, a small sample of code for reading the output value of the sensor is shown:

```
{
  float value_ldr = 0;
  SensorAgrv20.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_LDR);
  delay(100);  //waiting for the stabilization of the power supply
  value_ldr = SensorAgrv20.readValue(SENS_AGR_LDR);
}
```

You can find a complete example code for reading the luminosity sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-4-ldr-sensor-reading

4.5.3. Socket

In figure 28 we can see highlighted the socket upon which the LDR sensor must be placed. Since this sensor behaves as a simple resistor, polarity should not be taken into account when connecting it. More information about the sockets for casing applications can be found in section "Sockets for Casing".

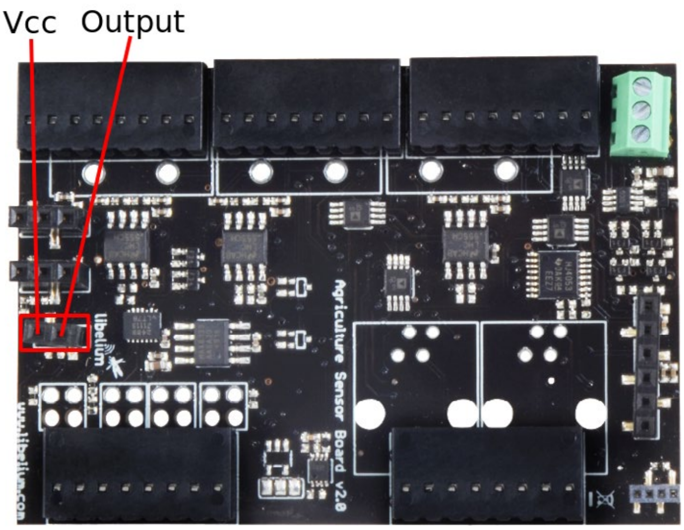


Figure 28: Image of the socket for the LDR sensor

4.6. Humidity+Temperature Sensor (SHT75)

4.6.1. Specifications

Power supply: 2.4 ~ 5.5V
Minimum consumption (sleep): 2µW
Consumption (measurement): 3mW
Average consumption: 90µW
Communication: Digital (two wire interface)
Storage temperature: 10 ~ 50ºC (0 ~ 80ºC maximum)
Storage humidity: 20 ~ 60%RH

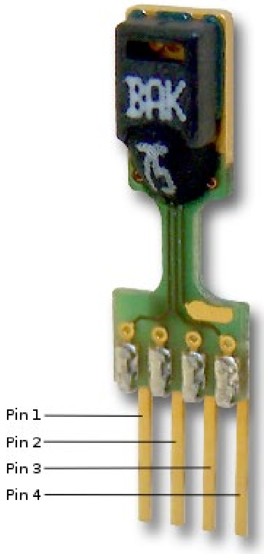


Figure 29: SHT75 sensor

Temperature:

Measurement range: -40ºC ~ +123.8ºC
Resolution: 0.04ºC (minimum), 0.01ºC (typical)
Accuracy: ±0.4ºC (range 0ºC ~ +70ºC), ±4ºC (range -40 ~ +125ºC)
Repeatability: ±0.1ºC
Response time (minimum): 5 seconds (63% of the response)
Response time (maximum): 30 seconds (63% of the response)

Humidity:

Measurement range: 0 ~ 100%RH
Resolution: 0.4%RH (minimum), 0.05%RH (typical)
Accuracy: ±1.8%RH
Repeatability: ±0.1%RH
Response time: 8 seconds

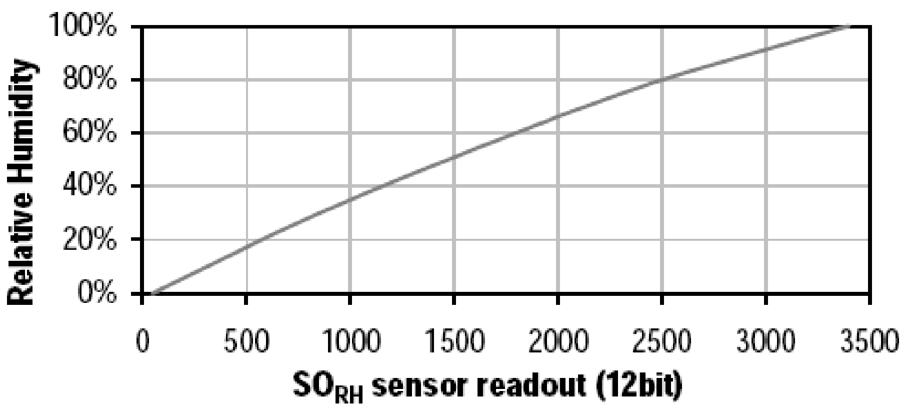


Figure 30: Graph of the SHT75 sensor output with respect to relative humidity, taken from the Sensirion sensor's data sheet

4.6.2. Measurement Process

The SHT75 sensor by Sensirion incorporates a capacitive sensor for environmental relative humidity and a band gap sensor for environmental temperature in the same package that permit to measure accurately both parameters. The sensor output is read through two wires (data and clock signals, connected to DIGITAL6 and DIGITAL8 pins respectively) following a protocol similar to the I2C bus (Inter- Integrated Circuit Bus). This protocol has been implemented in the library of the board, so the sensor can be read using the commands specifically designed to that function. The function of the library readValue returns the temperature value in Celsius degree (ºC) and the humidity value in relative humidity percentage (%RH). It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

Reading code:

```
{
  float value_temperature = 0;
  float value_humidity = 0;
  SensorAgrv20.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_SENSIRION);
  delay(10000);//waiting for the sensor's response time
  value_temperature = SensorAgrv20.readValue(SENS_AGR_SENSIRION, SENSIRION_TEMP);
  delay(100);  //the sensor sleeps until the next measurement
  value_humidity = SensorAgrv20.readValue(SENS_AGR_SENSIRION, SENSIRION_HUM);
}
```

You can find a complete example code for reading the humidity and temperature sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-6-digital-temperature-and-humidity-sensor-reading

4.6.3. Socket

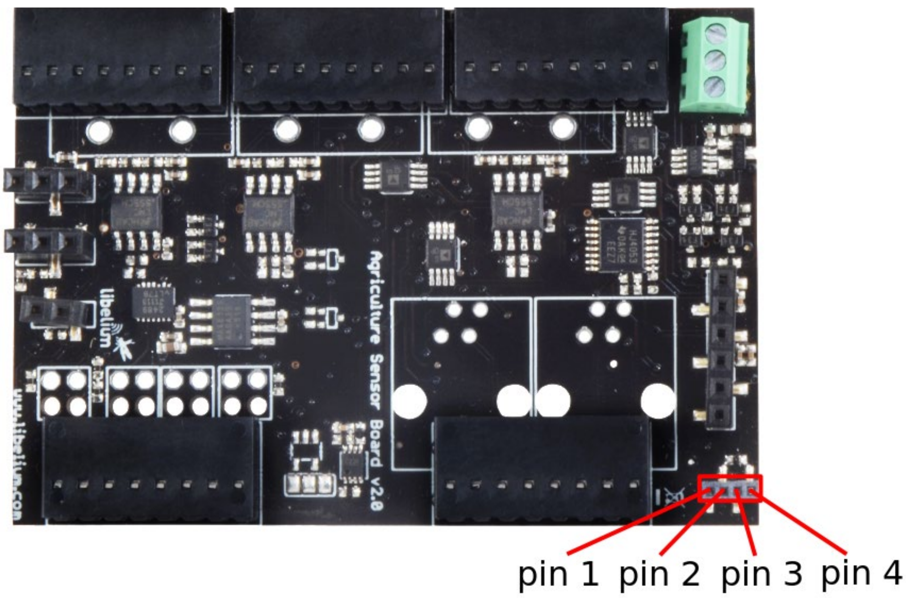


Figure 31: Image of the socket for the SHT75 sensor

The socket consists of a four female pin, 1.27mm pitch strip that connects the sensor to its digital outputs (DIGITAL8 and DIGITAL6 of Waspmote) ground and 3.3V power supply, regulated through a digital switch controlled by the DIGITAL5 pin of the mote. More information about the sockets for casing applications can be found in section "Sockets for Casing".

4.7. Soil Moisture Sensor (Watermark)

4.7.1. Specifications

Measurement range: 0 ~ 200cb
Frequency Range: 50 ~ 10000Hz approximately
Diameter: 22mm
Length: 76mm
Terminals: AWG 20

Figure 32: Watermark sensor

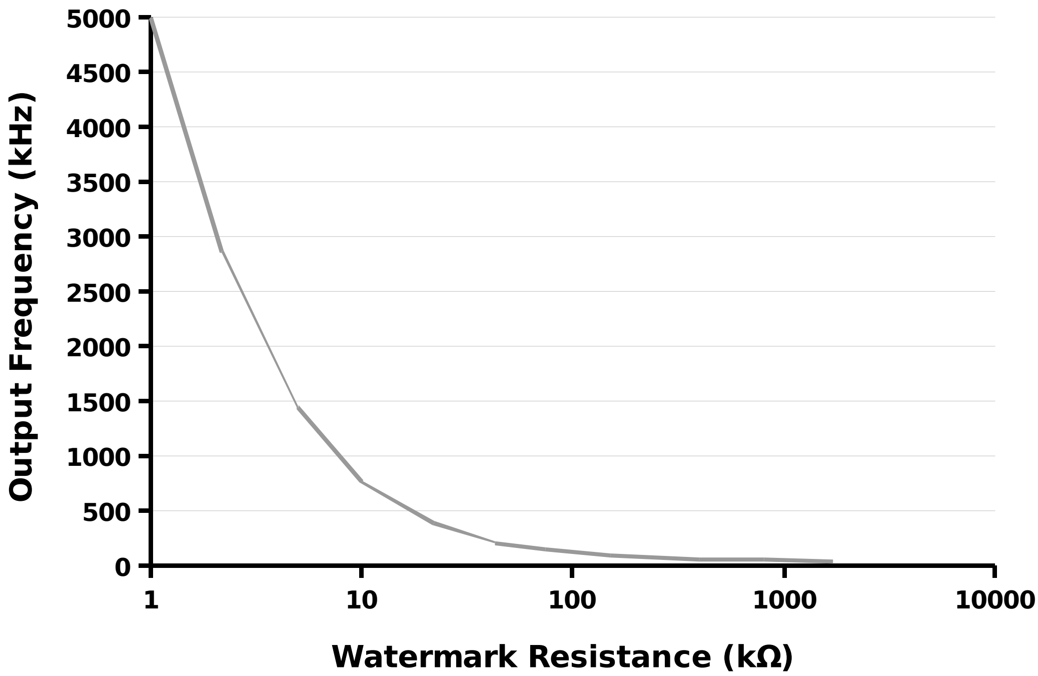


Figure 33: Output frequency of the Watermark sensor circuit with respect to the resistance of the sensor

4.7.2. Measurement Process

The Watermark sensor by Irrometer is a resistive type sensor consisting of two electrodes highly resistant to corrosion embedded in a granular matrix below a gypsum wafer. The resistance value of the sensor is proportional to the soil water tension, a parameter dependent on moisture that reflects the pressure needed to extract the water from the ground. The function of the library `readValue` returns the frequency output of the sensor's adaptation circuit in Herzs (Hz), for more information about the conversion into soil water tension look at Appendix 1 in this technical guide. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

Reading Code:

```
{
 float value_soil_humidity_1 = 0;
 float value_soil_humidity_2 = 0;
 float value_soil_humidity_3 = 0;
 SensorAgrv20.ON();
 SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_WATERMARK_1);
 delay(100);  //waiting for the stabilization of the power supply
 value_soil_humidity_1 = SensorAgrv20.readValue(SENS_AGR_WATERMARK_1);
 value_soil_humidity_2 = SensorAgrv20.readValue(SENS_AGR_WATERMARK_2);
 value_soil_humidity_3 = SensorAgrv20.readValue(SENS_AGR_WATERMARK_3);
}
```

You can find a complete example code for reading the Watermark sensor (Watermark 1 in this case) in the following link:

http://www.libelium.com/development/waspmote/examples/ag-11-watermark-sensor-reading

4.7.3. Socket

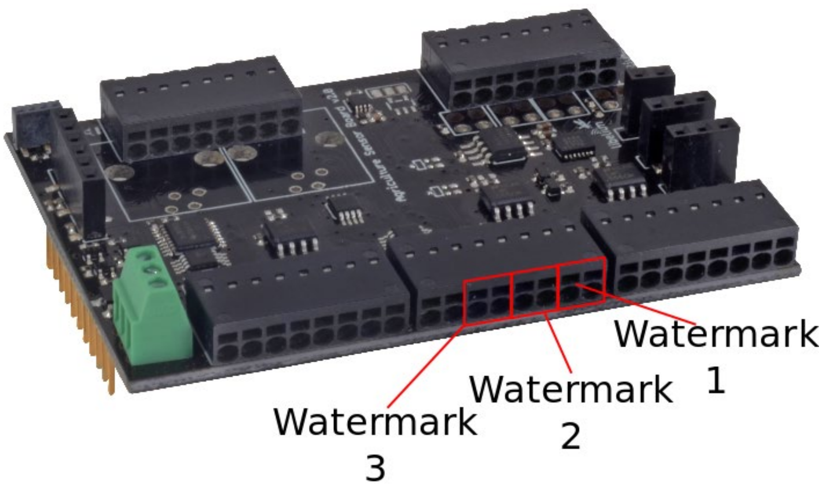


Figure 34: Image of the socket for the Watermark sensor

Three sockets for Watermark sensors have been placed in the agriculture board (marked in the image in figure 34) and the electronics necessary for powering and signal conditioning, so that the soil moisture can be measured at three different depths. Each of this outputs is carried to the input of a multiplexer controlled by the digital output `DIGITAL3`. These multiplexers permit the selection of two groups of sensors, the three Watermarks or the humidity, temperature and leaf wetness sensors, when reading the analog inputs `ANALOG2`, `ANALOG4` and `ANALOG6` of the Waspmote.

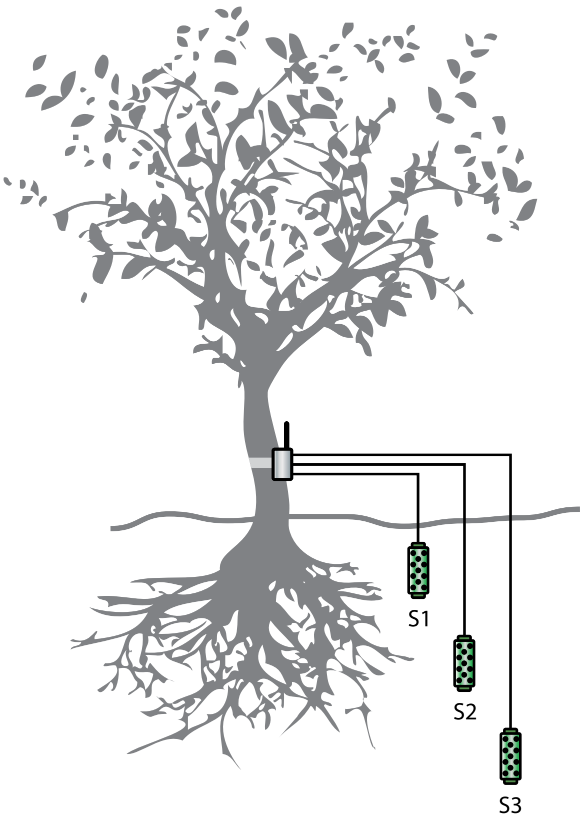


Figure 35: Illustration of the three Watermark sensors placed at different depths

4.8. Soil Temperature Sensor (PT-1000)

4.8.1. Specifications

Measurement range: -50 ~ 300ºC
Accuracy: DIN EN 60751
Resistance (0ºC): 1000Ω
Diameter: 6mm
Length: 40mm
Cable: 2m

Figure 36: PT-1000 sensor

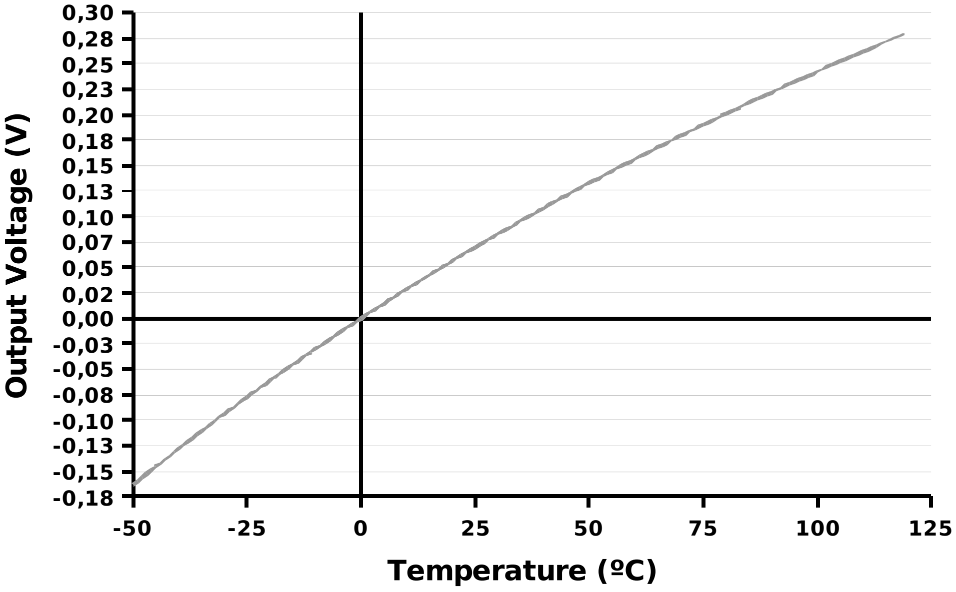


Figure 37: Output voltage of the PT1000 sensor with respect to temperature

4.8.2. Measurement Process

The resistance of the PT1000 sensor varies between approximately 920Ω and 1200Ω in the range considered useful in agriculture applications (-20 ~ 50ºC approximately), which results in too low variations of voltage at significant changes of temperature for the resolution of the Waspmote's analog-to-digital converter. The function of the library `readValue` returns the temperature value in Celsius degree (ºC). The power supplies required by the sensor, both 3.3V and 5V, are controlled through a digital switch that can be opened or closed via software activating the digital pin `DIGITAL5`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

This sensor only is included in the PRO version.

Reading code:

```
{
 float value_PT1000 = 0;
 SensorAgrv20.ON();
 SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_PT1000);
 delay(100);  //waiting for the stabilization of the power supply
 value_PT1000 = SensorAgrv20.readValue(SENS_AGR_PT1000);
}
```

You can find a complete example code for reading the PT1000 sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-10-pt1000-sensor-reading

4.8.3. Socket

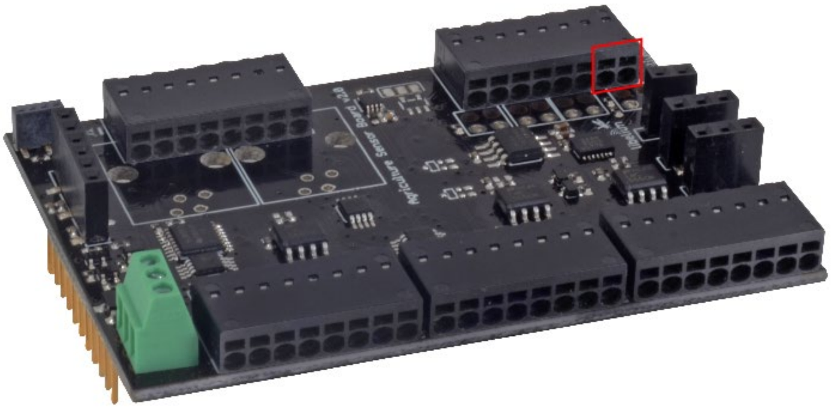


Figure 38: Image of the socket for the PT-1000 sensor

The sensor must be connected to its adaptation stage through a 2,54mm pitch. We can see an image with the two pins of this socket corresponding to the sensor in figure 38.

4.9. Trunk Diameter Dendrometer (Ecomatik DC2)

4.9.1. Specifications

Trunk/branch diameter: From 2 cm
Accuracy: ±2µm
Temperature coefficient: <0.1µm/K
Linearity: <2%
Operation temperature: -30 ~ 40ºC
Operation humidity: 0 ~ 100%RH
Cable length: 2m
Output range: 0 ~ 20kΩ
Range of the sensor: Function of the size of the tree

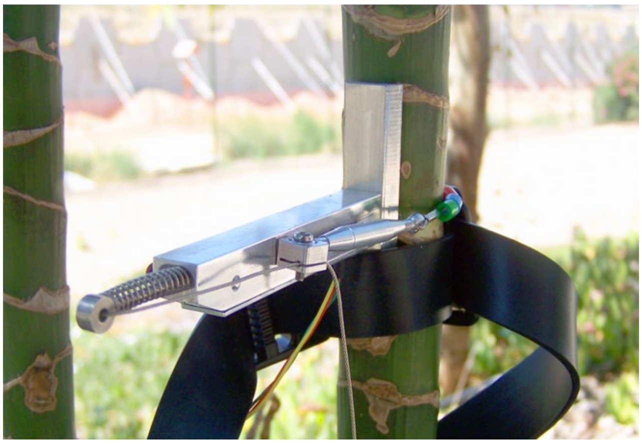


Figure 39: Ecomatik DC2 sensor

Tree Diameter (cm)	Measuring Range in Circumference (mm)	Measuring Range in Diameter (mm)
10	31.25	9.94
40	22.99	7.31
100	16.58	5.27

4.9.2. Measurement Process

The operation of the three Ecomatik dendrometers, DC2, DD and DF, is based on the variation of an internal resistance with the pressure that the growing of the trunk, stem, branch or fruit exerts on the sensor. The circuit permits the reading of that resistance in a full bridge configuration through a 16 bits analog-to-digital converter whose reference is provided by a high precision 3V voltage reference in order to acquire the most accurate and stable measurements possible. The reading of the converter, shared with the PT1000 temperature sensor, will be carried out through the I2C bus using the functions of the library `SensorAgrv20` explained in chapter "API" about this API library, returning the read value in millimeters (mm). The 3.3V and 5V power supplies of the devices may be cut off or connected via the digital switch controlled by the digital pin `DIGITAL5`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

This sensor only is included in the PRO version.

Reading code:

```
{
 float value_dendrometer = 0;
 SensorAgrv20.ON();
 SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_DENDROMETER);
 delay(100);  //waiting for the stabilization of the power supply
 value_dendrometer = SensorAgrv20.readValue(SENS_AGR_DENDROMETER);
}
```

You can find a complete example code for reading the dendrometer sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-9-dendrometer-sensor-reading

4.9.3. Socket

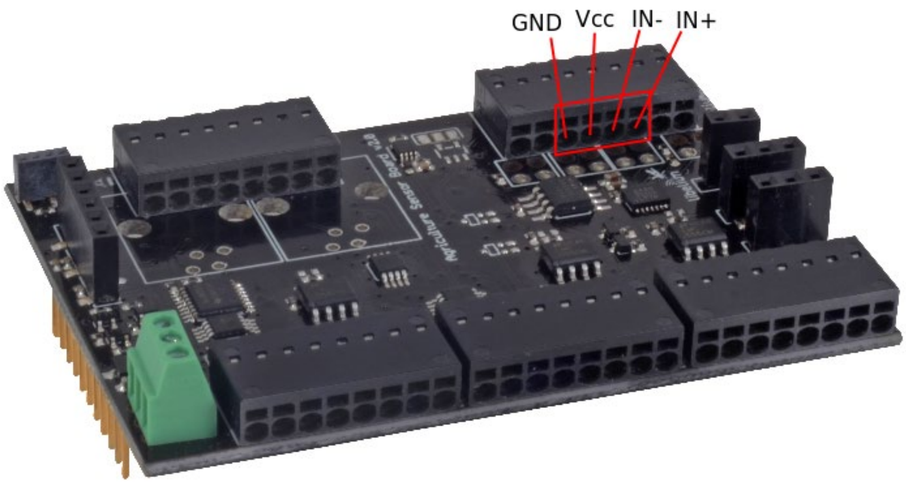


Figure 40: Image of the socket for the dendrometers

Any of the three dendrometers available may be connected to Waspmote through the two 2.54mm pitch terminal blocks marked in the image in figure 40. The pins corresponding to the sensor in this terminal block (highlighted in figure 40) provide the sensor with connection to ground, power supply and to the analog-to-digital converter differential input.

4.10. Stem Diameter Dendrometer (Ecomatik DD)

4.10.1. Specifications

Stem/branch diameter: 0 ~ 20cm
Range of the sensor: 11mm
Output range: 0 ~ 20kΩ
Accuracy: ±2µm
Temperature coefficient: <0.1µm/K
Operation temperature: -30 ~ 40ºC
Operation humidity: 0 ~ 100%RH
Cable length: 2m

Figure 41: Ecomatik DD sensor

4.10.2. Measurement Process

The operation of the three Ecomatik dendrometers, DC2, DD and DF, is based on the variation of an internal resistance with the pressure that the growing of the trunk, stem, branch or fruit exerts on the sensor. The circuit permits the reading of that resistance in a full bridge configuration through a 16 bits analog-to-digital converter whose reference is provided by a high precision 3V voltage reference in order to acquire the most accurate and stable measurements possible. The reading of the converter, shared with the PT1000 temperature sensor, will be carried out through the I2C bus using the functions of the library `SensorAgrv20` explained in chapter "API" about this API library, returning the read value in millimeters (mm). The 3.3V and 5V power supplies of the devices may be cut off or connected via the digital switch controlled by the digital pin `DIGITAL5`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

This sensor only is included in the PRO version.

Reading code:

```
{
 float value_dendrometer = 0;
 SensorAgrv20.ON();
 SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_DENDROMETER);
 delay(100);  //waiting for the stabilization of the power supply
 value_dendrometer = SensorAgrv20.readValue(SENS_AGR_DENDROMETER);
}
```

You can find a complete example code for reading the dendrometer sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-9-dendrometer-sensor-reading

4.10.3. Socket

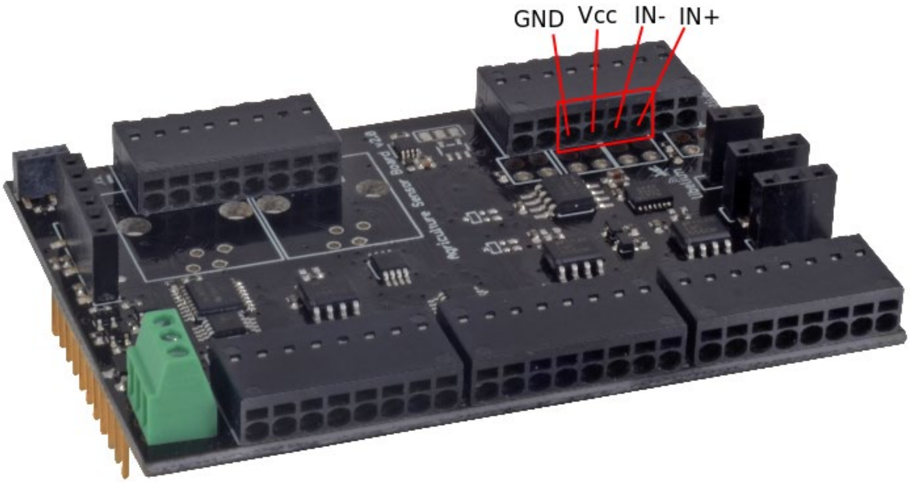


Figure 42: Image of the socket for the dendrometers

Any of the three dendrometers available may be connected to Waspmote through the two 2.54mm pitch terminal blocks marked in the image in figure 42. The pins corresponding to the sensor in this terminal block (highlighted in figure 42) provide the sensor with connection to ground, power supply and to the analog-to-digital converter differential input.

4.11. Fruit Diameter Dendrometer (Ecomatik DF)

4.11.1. Specifications

Fruit diameter: 0 ~ 11cm
Range of the sensor: 11mm
Output range: 0 ~ 20kΩ
Accuracy: ±2μm
Temperature coefficient: <0.1μm/K
Operation temperature: -30 ~ 40ºC
Operation humidity: 0 ~ 100%RH
Cable length: 2m

Figure 43: Ecomatik DF sensor

4.11.2. Measurement Process

The operation of the three Ecomatik dendrometers, DC2, DD and DF, is based on the variation of an internal resistance with the pressure that the growing of the trunk, stem, branch or fruit exerts on the sensor. The circuit permits the reading of that resistance in a full bridge configuration through a 16 bits analog-to-digital converter whose reference is provided by a high precision 3V voltage reference in order to acquire the most accurate and stable measurements possible. The reading of the converter, shared with the PT1000 temperature sensor, will be carried out through the I2C bus using the functions of the library `SensorAgrv20` explained in chapter "API" about this API library, returning the read value in milimeters (mm). The 3.3V and 5V power supplies of the devices may be cut off or connected via the digital switch controlled by the digital pin `DIGITAL5`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

This sensor only is included in the PRO version.

Reading code:

```
{
  float value_dendrometer = 0;
  SensorAgrv20.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_DENDROMETER);
  delay(100);  //waiting for the stabilization of the power supply
  value_dendrometer = SensorAgrv20.readValue(SENS_AGR_DENDROMETER);
}
```

You can find a complete example code for reading the dendrometer sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-9-dendrometer-sensor-reading

4.11.3. Socket

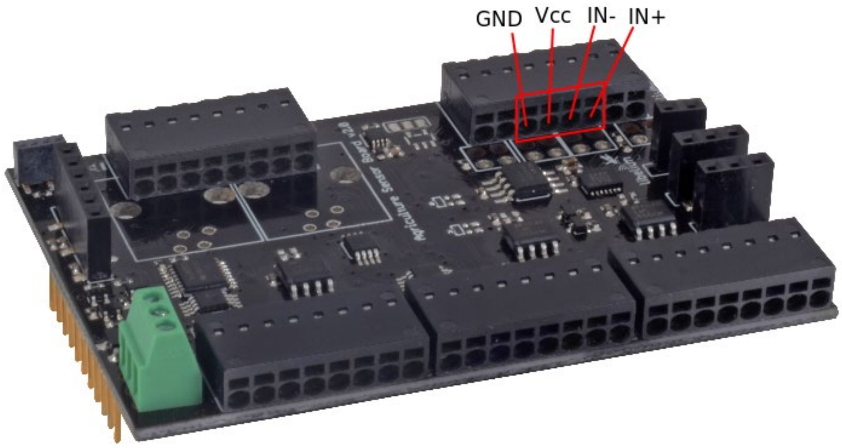


Figure 44: Image of the socket for the dendrometers

Any of the three dendrometers available may be connected to Waspmote through the two 2.54mm pitch terminal blocks marked in the image in figure 44. The pins corresponding to the sensor in this terminal block (highlighted in figure 44) provide the sensor with connection to ground, power supply and to the analog-to-digital converter differential input.

4.12. Solar Radiation Sensor - PAR (SQ-110)

4.12.1. Specifications

Responsivity: 0.200mV / $\mu mol \cdot m^{-2}s^{-1}$
Maximum radiation output: 400mV (2000$\mu mol \cdot m^{-2}s^{-1}$)
Lineal range: 1000mV (5000$\mu mol \cdot m^{-2}s^{-1}$)
Sensibility: 5.00$\mu mol \cdot m^{-2}s^{-1}$/mV
Spectral range: 400 ~ 700nm
Accuracy: ±5%
Repeatability: ±1%
Diameter: 2.4cm
Height: 2.75cm
Cable length: 3m
Operation temperature: -40 ~ 55ºC
Operation humidity: 0 ~ 100%RH

Figure 45: SQ-110 sensor

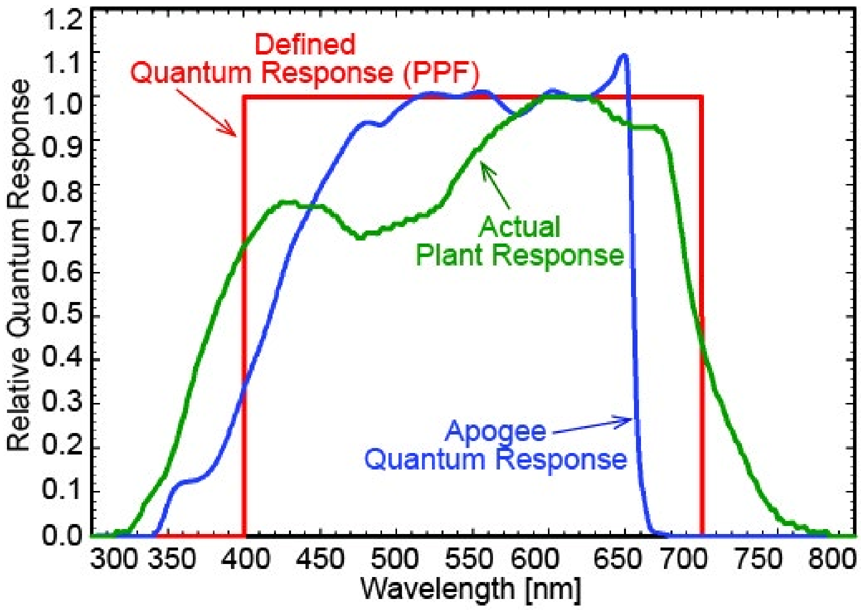


Figure 46: Graph of the spectral response of the SQ-110 sensor compared to the photosynthetic response of a plant

4.12.2. Measurement Process

The SQ-110 sensor, specifically calibrated for the detection of solar radiation, provides at its output a voltage proportional to the intensity of the light in the visible range of the spectrum, a key parameter in photosynthesis processes. It presents a maximum output of 400mV under maximum radiation conditions and a sensitivity of 5.00µmol·m^{-2}s^{-1}/mV. In order to improve the accuracy of the reading, this is carried out through a 16 bits analog-to-digital converter that communicates with the microprocessor of the mote through the I2C. It can be configured or read using the functions implemented in the API library for the Agriculture 2.0 Board (`SensorAgrV20`). The 5V power supply of this stage is controlled through a digital switch that can be activated and deactivated using the digital pin `DIGITAL5`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

This sensor only is included in the PRO version.

Reading code

```
{
 float value_radiation = 0;
 SensorAgrv20.ON();
 SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_RADIATION);
 delay(100);    //waiting for the stabilization of the power supply
 value_radiation = SensorAgrv20.readValue(SENS_AGR_RADIATION);
 //Conversion into µmol·m-2s-1
 value_radiation = value_radiation / 0.0002;
}
```

You can find a complete example code for reading the SQ-110 sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-8-photosynthetic-solar-radiation-sensor-reading

4.12.3. Socket

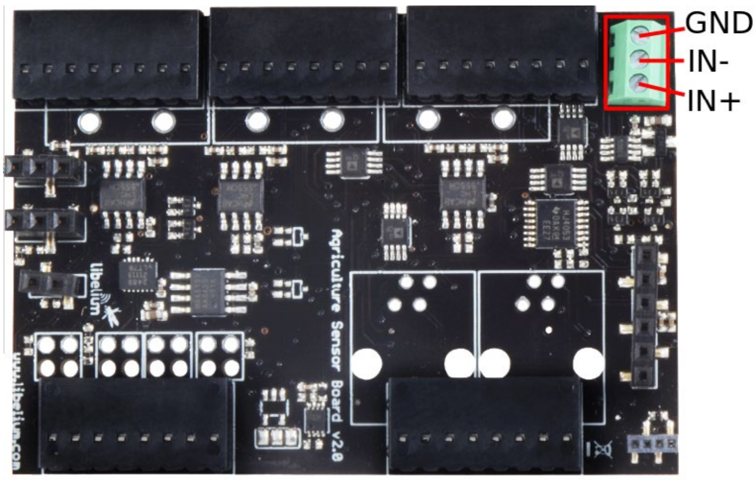


Figure 47: Image of the socket for the SQ-110 sensor

This sensor is connected to the analog-to-digital converter through the three pins of the 2.54mm pitch terminal block marked in the image in figure 47.

4.13. Ultraviolet Radiation Sensor (SU-100)

4.13.1. Specifications

Responsivity: 0.15mV / $\mu mol \cdot m^{-2}s^{-1}$
Maximum radiation output: 26mV (170$\mu mol \cdot m^{-2}s^{-1}$)
Lineal range: 60mV (400$\mu mol \cdot m^{-2}s^{-1}$)
Sensibility: 6.5$\mu mol \cdot m^{-2}s^{-1}$/mV
Spectral range: 250 ~ 400nm
Accuracy: ±10%
Repeatability: ±1%
Diameter: 2.4cm
Height: 2.75cm
Cable length: 3m
Operation humidity: 0 ~ 100%RH

Figure 48: SU-100 sensor

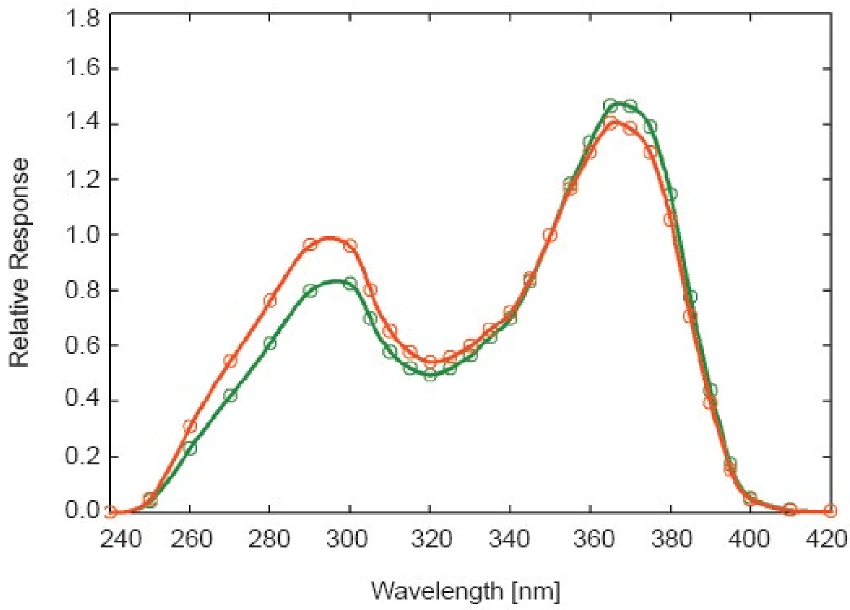


Figure 49: Graph of the spectral response of the SU-100 sensor compared to the photosynthetic response of a plant

4.13.2. Measurement Process

The SU-100 sensor, complementary to the SQ-110 sensor, provides at its output a voltage proportional to the intensity of the light in the ultraviolet range of the spectrum. It presents a maximum output of 26mV under maximum radiation conditions and a sensitivity of 6.5µmol·m^{-2}s^{-1}/mV. This sensor is read by the mote through the same 16 bits analog-to-digital converter used with the SQ-110 sensor. It can be configured or read using the functions implemented in the API library for the Agriculture 2.0 Board (`SensorAgrv20`). The 5V power supply of this stage is controlled through a digital switch that can be activated and deactivated using the digital pin `DIGITAL5`. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

This sensor only is included in the PRO version.

Reading code

```
{
 float value_radiation = 0;
 SensorAgrv20.ON();
 SensorAgrv20.setSensorMode(SENS_AGR_RADIATION, SENS_ON);
 delay(100);  //waiting for the stabilization of the power supply
 value_radiation = SensorAgrv20.readValue(SENS_AGR_RADIATION);
 //Conversion into µmol·m-2s-1
 value_radiation = value_radiation / 0.00015;
}
```

You can find a complete example code for reading the SU-100 sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-7-ultraviolet-solar-radiation-sensor-reading

4.13.3. Socket

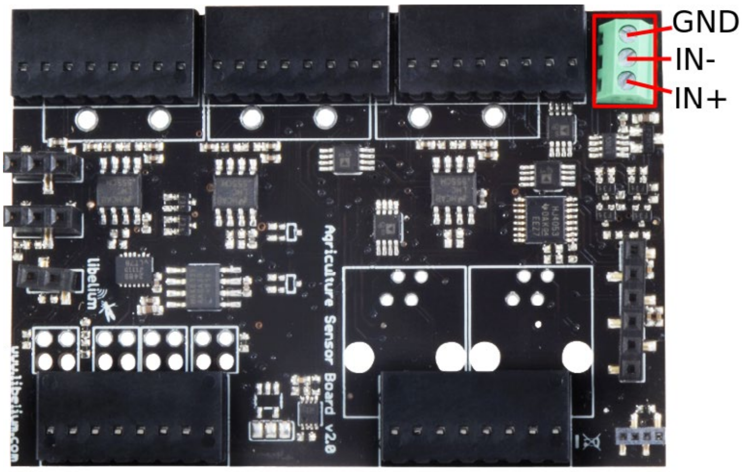


Figure 50: Image of the socket for the SU-100 sensor

This sensor is connected to the analog-to-digital converter through the three pins of the 2.54mm pitch terminal block marked in the image in figure 50.

4.14. Weather Station

The weather station consists of three different sensors, described in detail below: a wind vane, an anemometer and a pluviometer. It connects to Waspmote through six wires that must be connected to the terminal block shown in figures 53 and 55, being the anemometer connected to the vane through an RJ11 socket.

Figure 51: Image of the Weather Station

4.14.1. Anemometer

4.14.1.1. Specifications

Sensitivity: 2.4km/h / turn
Wind Speed Range: 0 ~ 240km/h
Height: 7.1 cm
Arm length: 8.9 cm
Connector: RJ11

Figure 52: Anemometer

4.14.1.2. Measurement Process

The anemometer chosen for Waspmote consists of a Reed switch normally open that closes for a short period of time when the arms of the anemometer complete a turn, so the output is a digital signal whose frequency will be proportional to the wind speed. That signal can be read from the digital pin `DIGITAL2` of the Waspmote. The function of the library `readValue` returns the wind speed value in kilometers per hour (km/h). The power of the sensor and of the electronics around it may be connected or disconnected using a digital switch controlled by the `ANALOG1` pin. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming") .

Reading code:

```
{
  float value_anemometer = 0;
  SensorAgrv2.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_ANEMOMETER);
  delay(100);  //waiting for the stabilization of the power supply
  value_anemometer = SensorAgrv20.readValue(SENS_AGR_ANEMOMETER);
}
```

You can find a complete example code for reading the whole weather station data sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-12-weather-station-sensor-reading

4.14.1.3. Socket

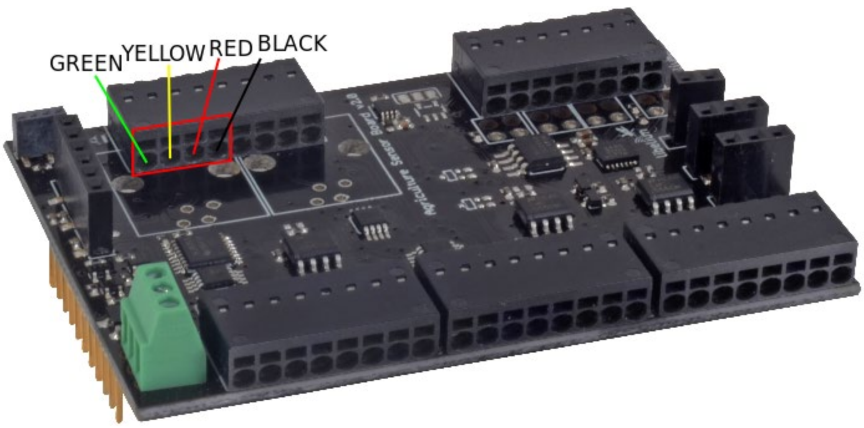


Figure 53: Image of the connector for the anemometer

The way to connect the anemometer to the Agriculture 2.0 Board is through the vane: the anemometer's cable must be plugged into the socket that can be found on the base of the vane.

4.14.2. Wind Vane

4.14.2.1. Specifications

Figure 54: Wind Vane

Height: 8.9 cm
Length: 17.8 cm
Maximum accuracy: 22.5º
Resistance range: 688Ω ~ 120kΩ

4.14.2.2. Measurement Process

The wind vane consists of a basement that turns freely on a platform endowed with a net of eight resistances connected to eight switches that are normally and are closed (one or two) when a magnet in the basement acts on them, which permits us to distinguish up to 16 different positions (the equivalent to a resolution of 22.5º). The equivalent resistance of the wind vane, along with a 10kΩ resistance, form a voltage divider, powered at 3.3V through a digital switch controlled by the `ANALOG1` pin, whose output can be measured in the analog input `ANALOG5`. The function of the library `readValue` also stores in variable `vane_direction` an 8 bits value which corresponds with an identifier of the pointing direction. It is highly recommended to switch off this sensor in order to minimize the global consumption of the board (you can find more information about the consumption of the board and how to handle it in chapter "Board configuration and programming").

Below, a table with the different values that the equivalent resistance of the wind vane may take is shown, along with the direction corresponding to each value:

Direction (Degrees)	Resistance (kΩ)	Voltage (V)	Identifier
0	33	2.53	SENS_AGR_VANE_N (0)
22.5	6.57	1.31	SENS_AGR_VANE_NNE (1)
45	8.2	1.49	SENS_AGR_VANE_NE (2)
67.5	0.891	0.27	SENS_AGR_VANE_ENE (3)
90	1	0.3	SENS_AGR_VANE_E (4)
112.5	0.688	0.21	SENS_AGR_VANE_ESE (5)
135	2.2	0.59	SENS_AGR_VANE_SE (6)
157.5	1.41	0.41	SENS_AGR_VANE_SSE (7)
180	3.9	0.92	SENS_AGR_VANE_S (8)
202.5	3.14	0.79	SENS_AGR_VANE_SSW (9)
225	16	2.03	SENS_AGR_VANE_SW (10)
247.5	14.12	1.93	SENS_AGR_VANE_WSW (11)
270	120	3.05	SENS_AGR_VANE_W (12)
292.5	42.12	2.67	SENS_AGR_VANE_WNW (13)
315	64.9	2.86	SENS_AGR_VANE_NW (14)
337.5	21.88	2.26	SENS_AGR_VANE_NNW (15)

Reading code:

```
{
  float value_vane = 0;
  SensorAgrv20.ON();
  SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_VANE);
  delay(100);  //waiting for the stabilization of the power supply
  value_vane = SensorAgrv20.readValue(SENS_AGR_VANE);
}
```

You can find a complete example code for reading the whole weather station data sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag-12-weather-station-sensor-reading

4.14.2.3. Socket

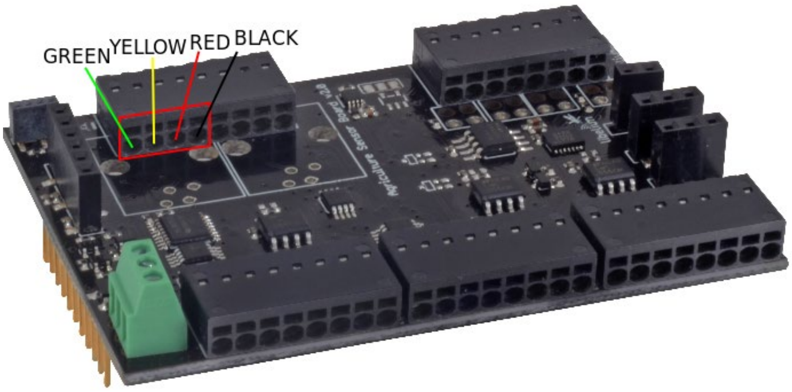


Figure 55: Image of the connector for the wind vane

The wind vane is connected to the board through four pins of a terminal block whose correspondence with the sensor's wire is shown in figure 55.

4.14.3. Pluviometer

4.14.3.1. Specifications

Height: 9.05 cm
Length: 23 cm
Bucket capacity: 0.28 mm of rain

Figure 56: Pluviometer

4.14.3.2. Measurement Process

The pluviometer consists of a small bucket that, once completely filled (0.28mm of water approximately), closes a switch, emptying automatically afterwards. The result is a digital signal whose frequency is proportional to the intensity of rainfall. The function of the library `readValue` returns the rain intensity in millimeters of rain per minute (mm/min). The sensor is connected directly to the Waspmote `DIGITAL4` digital input through a pull-up resistance and to the interruption pin `TXD1`, allowing the triggering of an interruption of the microprocessor when the start of the rain is detected. Since the consumption of this sensor, in absence of rain, is null, no elements of power control have been added for it.

Reading code:

```
{
  float value_pluviometer = 0;
  SensorAgrv20.ON();
  delay(100);  //waiting for the stabilization of the power supply
  value_pluviometer = SensorAgrv20.readValue(SENS_AGR_PLUVIOMETER);
}
```

You can find a complete example code for reading the whole weather station data sensor in the following link:

http://www.libelium.com/development/waspmote/examples/ag_12_weather_station_sensor_reading

4.14.3.3. Socket

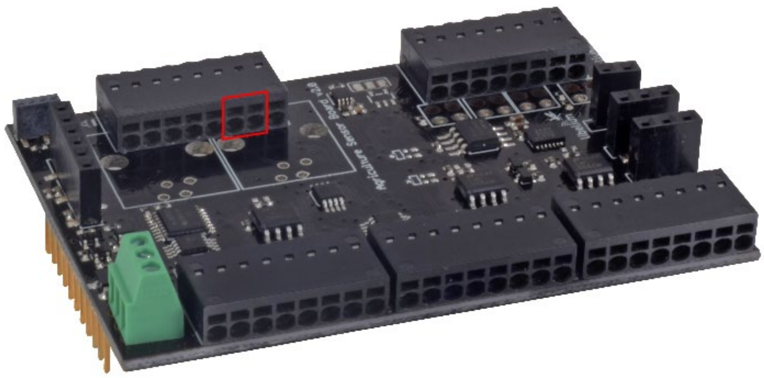


Figure 57: Image of the socket for the pluviometer

In the image in figure 57 we can see marked the two pins in the terminal block that correspond to the pluviometer (no polarity is required).

4.15. Integration of New Sensors

Those sensors that have been tested on the Agriculture 2.0 board by Libelium are detailed and recommended in this handbook. However, other different sensors, such as the 10HS, EC-5 and MPS-1 by Decagon, which present an output analogous to that of those previously described, as a resistance or an analog voltage, can be integrated in the board, taking into account their consumptions when developing the application and always respecting the mote's specifications (please refer to the "Hardware" section of the Waspmote technical guide).

4.16. Sockets for casing

In case the Agriculture 2.0 board is going to be used in an application that requires the use of a casing, such as an outdoors application, a series of sockets to facilitate the connection of the sensors through a probe has been disposed.

These sockets (PTSM from Phoenix Contact) allow to assemble the wires of the probe simply by pressing them into it. To remove the wire press the slot above the input pin and pull off the wire softly.

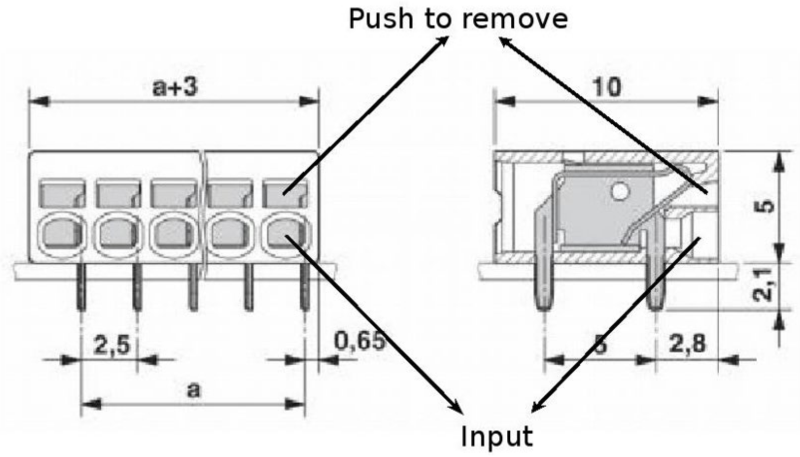


Figure 58: Diagram of a socket extracted from the Phoenix Contact data sheet

In the figure below an image of the board with the sockets in it and the correspondence between its inputs and the sensor's pins is shown.

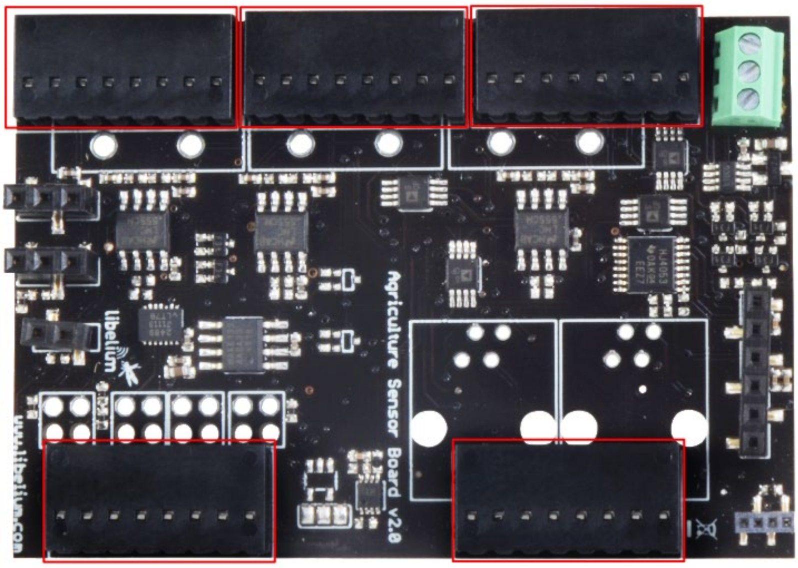


Figure 59: Image of the sockets for casing applications

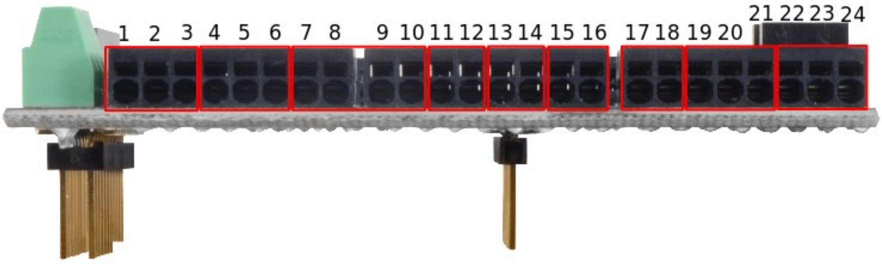


Figure 60: Image of the pin correspondence between the sockets and the sensors

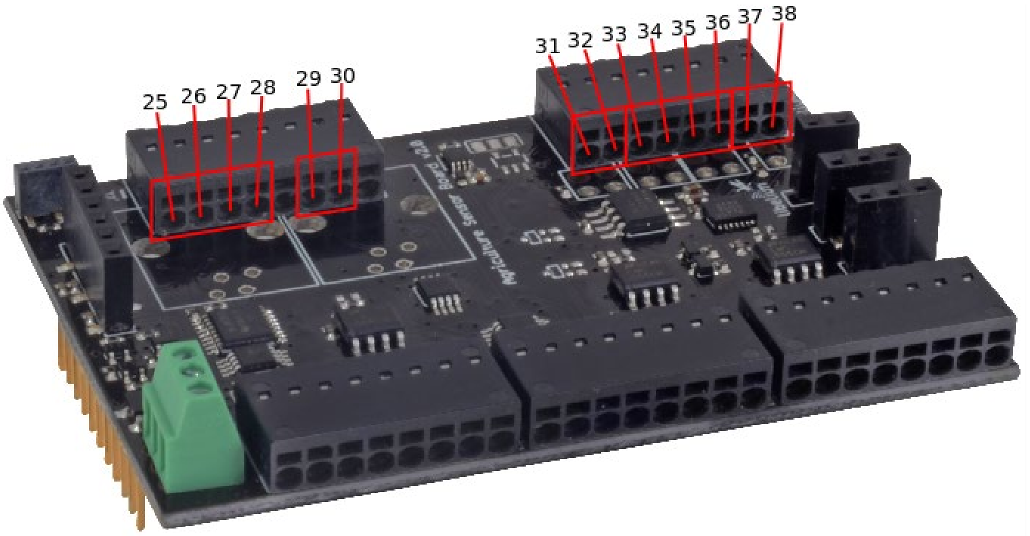


Figure 61: Image of the pin correspondence between the sockets and the sensors

Sensor	Pin	Function
Solar Radiation Sensor	1	GND
	2	Output -
	3	Output +
Atmospheric Pressure Sensor	4	Output
	5	GND
	6	VCC
Humidity and Temperature Sensor	7	DATA
	8	GND
	9	VCC
	10	CLK
Watermark 3	11	Output
	12	Output
Watermark 2	13	Output
	14	Output
Watermark 1	15	Output
	16	Output
Luminosity Sensor	17	VCC
	18	Output
Humidity Sensor	19	VCC
	20	Output
	21	GND
Temperature Sensor	22	VCC
	23	Output
	24	GND
Weather Station (Vane + Anemometer)	25	Vane Output (Green)
	26	Anemometer Output (Yellow)
	27	GND (Red)
	28	GND (Black)
Weather Station (Pluviometer)	29	Output
	30	GND
Leaf Wetness Sensor	31	GND
	32	Output
Dendrometer	33	GND
	34	VCC
	35	Output -
	36	Output +
PT1000	37	Output
	38	GND

5. Board configuration and programming

5.1. Hardware configuration

The Waspmote Agriculture 2.0 Board hardly requires a manual hardware configuration, since all the power control and sensor reading operations can be carried out digitally. It will be only necessary to ensure that the sensors are connected in the right way to their sockets for a proper measurement.

5.2. API

A library for the Agriculture 2.0 Board has been programmed, that added to the Waspmote IDE allows to manage in an easy way the resources of the board, the power supply, the start-up and reading of the sensors and the interruptions.

When using the Agriculture Sensor Board v20 on Waspmote PRO, remember it is mandatory to include the SensorAgrv20 library by introducing the next line at the beginning of the code:

```
#include <WaspSensorAgriculture_v20.h>
```

Next, the different functions that make up the library are described:

`SensorAgrv20.ON()`

Turns on the sensor board by activating the 3.3V and 5V supply lines.

`SensorAgrv20.OFF()`

Turns off the sensor board by cutting the 3.3V and 5V supply lines.

`SensorAgrV20.setBoardMode(MODE)`

This function is used to manage the power applied to the board. Assigning the value `SENS_ON` to the variable `MODE` activates the board's switches which allow the passage of the 3.3V and 5V supplies, while assigning the value `SENS_OFF` disconnects both switches cutting the power.

`SensorAgrV20.setSensorMode(MODE, TYPE)`

This function allows activation or deactivation of each of the sensors by assigning to the variable `MODE` the values `SENS_ON` (for turning the sensor on) or `SENS_OFF` (for turning the sensor off). The sensor to be managed is pointed through the variable `TYPE`.

Atmospheric pressure: `SENS_AGR_PRESSURE`
Watermark nº 1: `SENS_AGR_WATERMARK_1`
Watermark nº 2: `SENS_AGR_WATERMARK_2`
Watermark nº 3: `SENS_AGR_WATERMARK_3`
Anemometer: `SENS_AGR_ANEMOMETER`
Wind Vane: `SENS_AGR_VANE`
Dendrometer: `SENS_AGR_DENDROMETER`
PT1000: `SENS_AGR_PT1000`
Leaf wetness: `SENS_AGR_LEAF_WETNESS`
Temperature: `SENS_AGR_TEMPERATURE`
Humidity: `SENS_AGR_HUMIDITY`
Solar radiation: `SENS_AGR_RADIATION`
Temperature and humidity (Sensirion): `SENS_AGR_SENSIRION`

When turning on and off each of the sensors, take into account that many of them are grouped under the same solid state switch, so when you modify the state of any sensor of the group you will be also acting on all the other sensors associated to its control switch. You can find more information on how the sensors are grouped in chapter "Power control".

`SensorAgrv20.readValue(SENSOR, TYPE)`

The instruction `readValue` captures the output value of the sensor and stores it in floating point format in the variable to which it has been assigned. The sensor whose output we want to read is defined by the variable `SENSOR`, which can take the same values enumerated in the section about the `setSensorMode` function. The captured value is converted into the units corresponding to the sensor that has been read. The parameter `TYPE` is only necessary when the sensor to be read is the temperature and humidity SHT75 by Sensirion, for which it is required to point which of the two measurements is going to be made (`SENSIRION_TEMP` for temperature and `SENSIRION_HUM` for humidity).

`SensorAgrv20.attachPluvioInt(SENSOR)`

The `attachPluvioInt` function enables the interruptions generated by the pluviometer. Take into account that this sensor is permanently powered all the time the board is on so it will keep on triggering interruptions as long as they are enabled.

`SensorAgrv20.detachPluvioInt()`

Complementing the previous function, the aim of `detachPluvioInt` is to deactivate the interruptions generated by the pluviometer. After its execution the microcontroller will ignore any interruption which arrives from this sensor until the `attachPluvioInt` instruction is called again.

`SensorAgrv20.sleepAgr(TIME, OFFSET, MODE, OPTION, AGR_INTERRUPTION)`

The function `sleepAgr` is an adaptation of the function `deepSleep` in the library **WaspPWR.cpp** that allows to put the Waspmote to sleep turning the power of the board completely off or keeping the pluviometer circuits on if the interruptions of this sensor are going to be used to wake up the microprocessor. The parameters `TIME`, `OFFSET`, `MODE` and `OPTION` allow to define the time the mote will be in deep sleep mode before waking up with an RTC interruption and the modules that will be inactive during this time, like in the original function (look at the Waspmote technical guide and programming guide for more information). To activate the pluviometer interruptions the parameter `AGR_INTERRUPTION` must be assigned with the value `SENS_AGR_PLUVIOMETER` (remember not to deactivate the sensor board when defining the parameter `OPTION` for a correct operation of the interruptions).

A basic program to detect events from the board will present a similar structure to the following, subject to changes in dependence of the application:

1. The board is switched on using the function **`SensorAgrv20.setBoardMode`**.

2. Initialization of the RTC using **`RTC.ON`**.

3. Activation of the sensors to generate given interruptions using function **`SensorAgrv20.setSensorMode`**.

4. Put the mote to sleep with function **`SensorAgrv20.sleepAgr`**.

5. When the mote wakes up, disable interruptions from the board using function **`SensorAgrv20.detachPluvioInt`**.

6. Process the interruption:

- Turn on those inactive sensors to be read using function **`SensorAgrv20.setSensorMode`**.
- Take the measurements needed using function **`SensorAgrv20.readValue`**.
- Turn off the sensors that shall not generate an interruption with function **`SensorAgrv20.setSensorMode`**.
- Store or send via a radio module the gathered information.

7. Return to step 4 to enable interruptions and put the mote to sleep.

Below you can see a sample of code in which the board is activated and put to sleep for ten minutes with the interruptions of rainfall activated. When an interruption from the RTC or the sensor board comes in, the mote wakes up and the pluviometer, dendrometer, anemometer and temperature sensor are read in function of the kind of interruption arrived. The measurements taken are transmitted via XBee.

```
/* ------------Agriculture 2.0 Sensor Board example---------------

 www.Libelium.com
 */

// Inclusion of the Events Sensor Board v20 library
#include <WaspSensorAgr_v20.h>

// Inclusion of the Frame library
#include <WaspFrame.h>

// Inclusion of the XBee 802.15.4 library
#include <WaspXBee802.h>

// Pointer to an XBee packet structure
packetXBee* packet;

//Variable definition
float value_temperature = 0;
float value_dendrometer = 0;
float value_anemometer = 0;
float value_pluviometer = 0;

void setup()
{
  // Turn on the Agriculture 2.0 Sensor Board
  SensorAgrv20.ON();
  // Init the RTC
  RTC.ON();
  delay(100);
}

void loop()
{
  // Put the mote to sleep with pluviometer interruptions enabled
  SensorAgrv20.sleepAgr("00:00:10:00", RTC_OFFSET, RTC_ALM1_MODE1, UART0_OFF | UART1_OFF |
BAT_OFF, SENS_AGR_PLUVIOMETER);
  // Detach pluviometer interruptions
  SensorAgrv20.detachPluvioInt();

  // Create new frame (ASCII)
  frame.createFrame(ASCII,"Waspmote_Pro");

  // In case a pluviometer interruption arrived
  if(intFlag & PLV_INT)
  {
    // Read the pluviometer
    value_pluviometer = SensorAgrv20.readValue(SENS_AGR_PLUVIOMETER);

    // Add the value read to the frame composition
    frame.addSensor(SENSOR_PLV, value_pluviometer);

  }

  // In case an RTC interruption arrived
  if(intFlag & RTC_INT)
  {

    // Turn on the sensors
    SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_ANEMOMETER);
    SensorAgrv20.setSensorMode(SENS_ON, SENS_AGR_TEMPERATURE);
```

```
    delay(100);
    // Read the sensor values
    value_anemometer = SensorAgrv20.readValue(SENS_AGR_ANEMOMETER);
    value_temperature = SensorAgrv20.readValue(SENS_AGR_TEMPERATURE);
    value_dendrometer = SensorAgrv20.readValue(SENS_AGR_DENDROMETER);

    // Add the values read to the frame composition
    frame.addSensor(SENSOR_ANE, value_anemometer);
    frame.addSensor(SENSOR_TCA, value_temperature);
    frame.addSensor(SENSOR_SD, value_dendrometer);

  }

  // Init XBee
  xbee802.ON();
  // Set parameters to packet:
  packet=(packetXBee*) calloc(1,sizeof(packetXBee));
  packet->mode=BROADCAST;

  // Set destination XBee parameters to packet
  xbee802.setDestinationParams( packet, "000000000000FFFF", frame.buffer, frame.length);

  // Send XBee packet
  xbee802.sendXBee(packet);

  // Turn off the XBee Module
  xbee802.OFF();
  delay(100);

  // Clear the interruption flag
  clearIntFlag();
}
```

The files related to this sensor board are: **WaspSensorAgr_v20.cpp, WaspSensorAgr_v20.h**

They can be downloaded from: **http://www.libelium.com/development/waspmote/sdk_and_applications**

6. API Changelog

Function / File	Changelog	Version
`#include`	Remember to include the WaspSensorAgr_v20 library in the top of your pde	v.31 → v1.0
`SensorAgrv20.ON()`	New function to turn on the board	v.31 → v1.0
`SensorAgrv20.OFF()`	New function to turn off the board	v.31 → v1.0
`SensorAgrv20.vane_direction`	The type of the variable changes from uint16_t into uint8_t. Also the values it can take change (take a look at the section dedicated to the Vane)	v.31 → v1.0

7. Consumption

7.1. Power control

The electronics of the Agriculture 2.0 Board requires the **3.3V** power supply from the Waspmote regulator and the **5V** power supply from the DC-DC converter, both can be controlled from the microprocessor of the mote using the function `setBoardMode` of the API for turning it completely on or off.

Inside the board itself, the power of the sensors **may be managed separately** using the digital solid state switches system like in other sensor boards for the Waspmote platform, allowing to keep the board on with a minimum consumption of 106µA. This way, the 15 sensors that, as a maximum, can be connected simultaneously to the Agriculture 2.0 Board have been put together into four different groups following two criteria: the consumption of the sensors and the job their perform. The API functions that control the activation of the sensors, as the rest of the functions specifically created for this board, can be found in the library `SensorAgr20`, explained in chapter "API".

Watermark Group: The first of this groups is formed by **the three Watermark** sensors for soil humidity and electronic adaptation stages described in chapter "Socket" of the Watermark sensor section. The 3.3V power supply is connected to the sensors and the rest of the electronics through switch 2, that can be manipulated through the digital pin `DIGITAL1`.

Meteorology Group: The second group is formed by **the wind vane and the anemometer** of the weather station described in sections "Weather Station". Again, these sensors only need the 3.3V power supply, controlled by the pin `ANALOG1` (configured as a digital output pin) by the switch 4.

Low Consumption Group: The third group is formed by **all the low consumptions sensors**: the dendrometer, the temperature sensor, the humidity sensor, the luminosity sensor, the leaf wetness sensor, the soil temperature sensor, the solar radiation sensor and the humidity plus temperature sensor. The regulation of the power of these sensors requires three different integrated circuits (switches 1, 2 and 3), all of them controlled through the same digital pin (`DIGITAL5`). This group of sensors needs both the 5V and the 3.3V power supplies.

At last, the **atmospheric pressure sensor** MPX4115A is powered independently of the rest of the sensors owing to its large consumption. Its power supply (5V) is controlled also through the switch 3, but in this case its activation and deactivation is carried out by the digital pin `DIGITAL7`.

The **pluviometer** is kept powered all the time at 3.3V, except when the whole board is turned off from the Waspmote. The convenience of monitoring the rainfall all the time once the mote has been deployed in determined applications is the main reason for it, along with the fact that the consumption of this sensor in absence of rain is almost null.

7.2. Tables of consumption

In the following table the consumption of the board is shown, the constant minimum consumption (fixed by the permanently active components), the minimum consumption of the electronics included in each group formed by the switches (without sensors) and the individual consumptions of each of the sensors connected alone to the board (the total consumption of the board with a determined sensor will be calculated as the sum of the constant minimum consumption of the board plus the minimum consumption of the group to whom the sensor belongs plus the consumption of the sensor).

Remember that the board's power can be completely disconnected, reducing the consumption to zero, using the 3.3V and the 5V main switches disconnection command included in the library.

	Consumption
Minimum (Constant)	0µA
Weather Station group	0µA
Watermark Sensors group	1.4mA
Low Consumption group	1.8mA
Atmospheric Pressure sensor	112µA
Watermark (1 sensor)	<0.8mA
Watermark (2 sensors)	<1.5mA
Watermark (3 sensors)	<2.2mA
Anemometer	<400µA
Wind Vane	<300µA
Pluviometer	0µA (330µA in ON Pulse -10ms approximately-)
Humidity (808H5V5)	0.7mA
Temperature (MCP9700A)	6µA
Luminosity (LDR)	<360µA
SHT75	<1µA
Atmospheric Pressure (Sensor)	7mA
Solar Radiation sensor - PAR (SQ-110)	0µA
PT1000	1.5mA
Dendrometers	160µA
Leaf Wetness Sensor	<240µA

7.3. Low consumption mode

The Waspmote Agriculture 2.0 Board has been designed to have the least consumption possible. For this, the only recommendations which the user must try to follow are the following:

- **Switch off the sensor groups that are not to be used**
  Turn on only those sensors to be measured and only for the necessary time to acquire their output values.
- **Use the Waspmote low consumption mode**
  This board's library includes a command to put the mote into a low consumption mode. Use it during the time in which the mote is not carrying out any measurement and try to space them as much as your application allows you.
- **Do not connect sensors that are not going to be used**
  Since several sensors share the same power line, a sensor that is not going to be used connected to the board will entail an additional consumption, and so a shorter life of the battery.

8. Documentation changelog

- Added references to 3G/GPRS Board in section: Radio Interfaces.

9. Maintenance

- In this section, the term "Waspmote" encompasses both the Waspmote device itself as well as its modules and sensor boards.
- Take care with the handling of Waspmote, do not drop it, bang it or move it sharply.
- Avoid putting the devices in areas of high temperatures since the electronic components may be damaged.
- The antennas are lightly threaded to the connector; do not force them as this could damage the connectors.
- Do not use any type of paint for the device, which may damage the functioning of the connections and closure mechanisms.

10. Disposal and recycling

- In this section, the term "Waspmote" encompasses both the Waspmote device itself as well as its modules and sensor boards.
- When Waspmote reaches the end of its useful life, it must be taken to a recycling point for electronic equipment.
- The equipment has to be disposed on a selective waste collection system, different to that of urban solid waste. Please, dispose it properly.
- Your distributor will inform you about the most appropriate and environmentally friendly waste process for the used product and its packaging.

Appendix 1: Watermark sensor's interpretation reference

The next table shows the resistance value of the sensor at different soil water tension conditions (at a 75 Fahrenheit degree, equivalent to 23.8 Celsius degree):

Soil Water Tension (cbar)	Sensor Resistance (Ohms)
0	550
9	1000
10	1100
15	2000
35	6000
55	9200
75	12200
100	15575
200	28075

Figure 62: Table 1. Resistance values of the sensor in function of the soil water tension

This series may be approached by this equation:

Equation 1: $Rs = 137.5 \times TA + 550$

Where TA is the soil water tension expressed in centibars.

From the real values and the approximation we obtain the next graph of the sensor resistance versus the soil water tension.

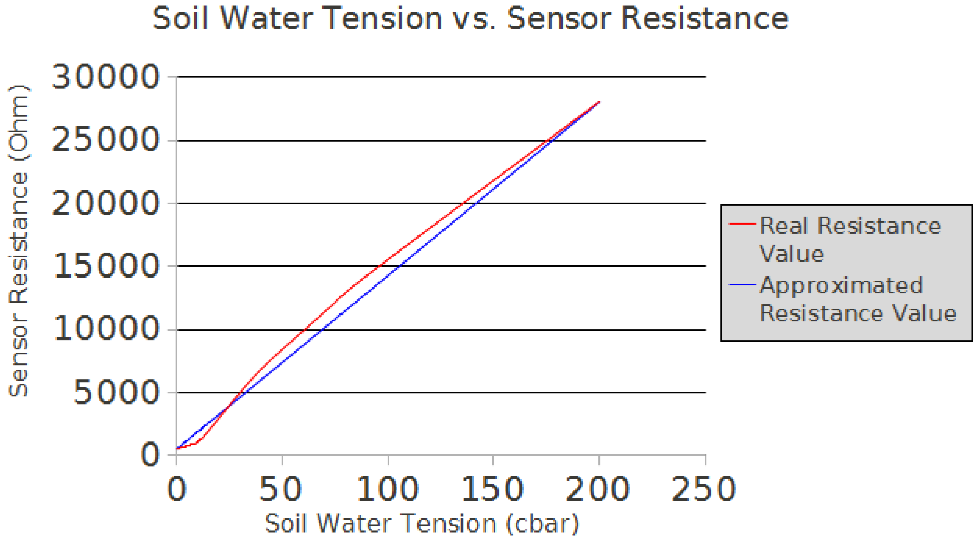


Figure 63: Resistance of the sensor in function of the soil water tension.

In the next figure we can see the frequency of the output of the adaptation circuit for the sensor, for the real resistance values and for the linearly approximated resistance values.

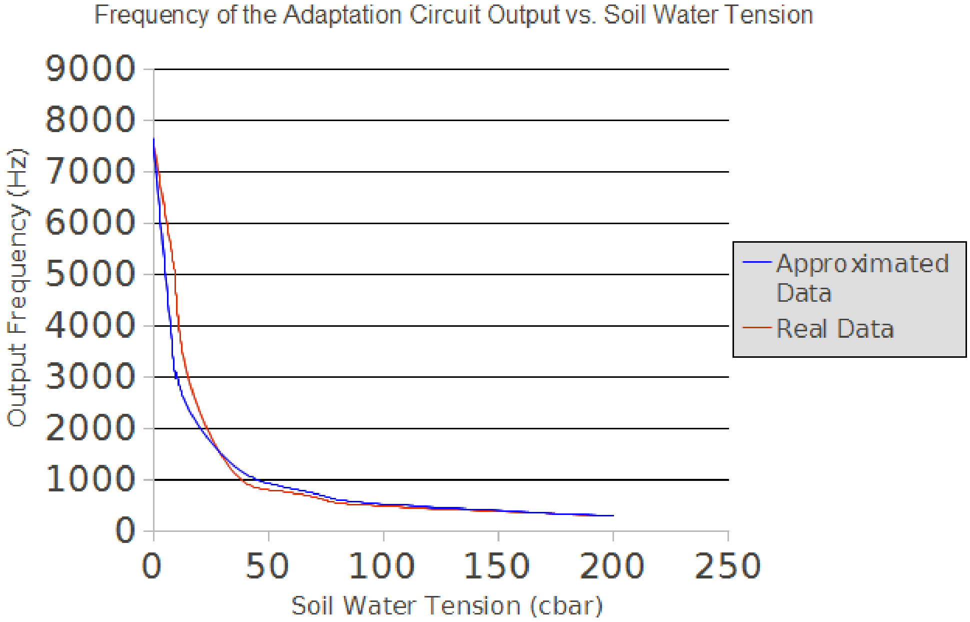


Figure 64: Output frequency of the adaptation circuit in function of the soil water tension.

The formula used to draw this graph, from the sensor resistance, is shown bellow:

Equation 2: $$F=\frac{(Rs+150390)}{(0.021\times Rs+8.19)}$$

Where F is the output frequency in Hz and R_S the sensor resistance in ohms.

If we substitute Equation 1 in Equation 2, we get the output frequency in function of the soil water tension:

Equation 3: $$F=\frac{(137.5\times TA+550+150390)}{(0.021\times(137.5\times TA+550)+8.19)}$$

Equation 4: $$F=\frac{(137.5\times TA+150940)}{(2.8875\times TA+19.74)}$$

We can see that the frequency output for the working range is between 300Hz (corresponding to the 200cbar of maximum soil water tension) and 7600Hz approximately for 0cbar measurement. It has been empirically checked that for very wet soils, bellow 10cbar, the behavior of different sensors is very variable, so calibration is highly recommended if accuracy under these conditions is needed.

To obtain the response of the sensor beyond this range, over the 200cbar, we must extrapolate those soil water tension values from the linear approximation obtained in equation 1. These sensors are not prepared for working under those conditions, so these graph must be only taken as a reference.

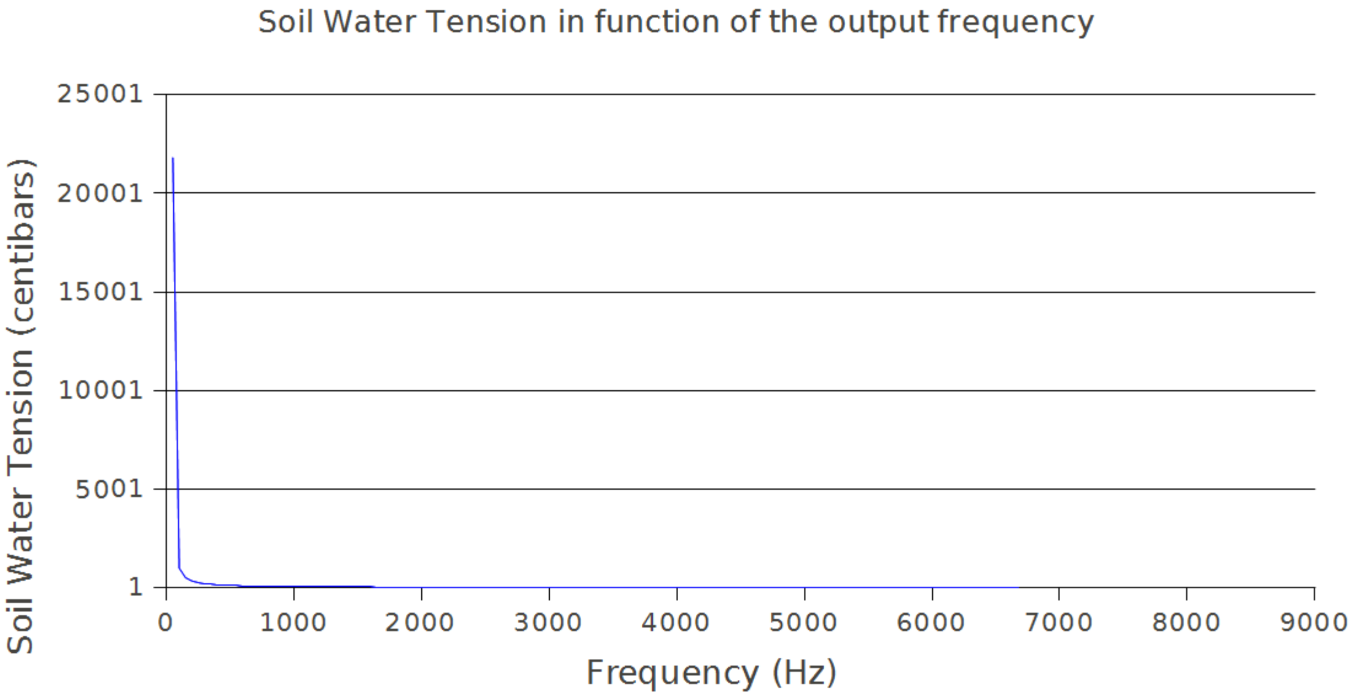


Figure 65: Soil water tension in function of the output frequency of the circuit.

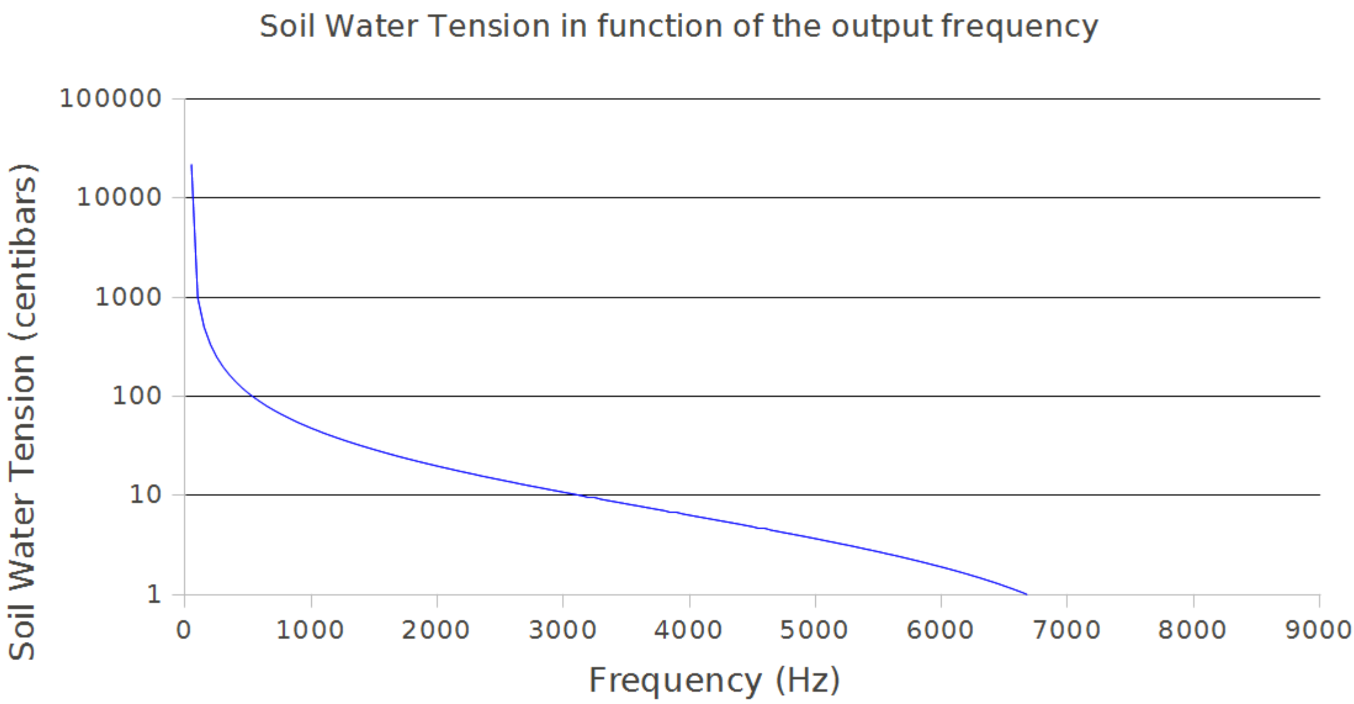


Figure 66: Soil water tension (in logarithmic scale) in function of the output frequency of the circuit.